ABSTRACT

ADITYA P. GOSWAMI. Implementation of Microwave Measurements using Novel Calibration Techniques. (Under the direction of Dr. Michael Steer.)

NetA (Network Analysis) tools for calibration of microwave measurements has been implemented. NetA contains calibration and de–embedding procedures as data analysis MATLAB routines. The Through Line method for calibration of two ports has been used and the NetA process flow has also been explained. Complex characteristic impedance of the micro-strip transmission line has been calculated using the ETRL (Enhanced TRL) technique. Results have been simulated using NetA tools. A LabVIEW Implementation of NetA has also been implemented so as to enhance the usability of NetA and also provide the capability of Real–time microwave calibration and de–embedding.

IMPLEMENTATION OF MICROWAVE MEASUREMENTS USING NOVEL CALIBRATION TECHNIQUES

by

Aditya P. Goswami

A thesis submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the Degree of
Master of Science

Computer Engineering

Raleigh

May 2003

APPROVED BY:

Chair of Advisory Committee

Dedication

This thesis is dedicated to my parents Pramod Gosai and Jyoti Gosai who have worked so hard and always given me the best education. They have raised me with a freedom of choice and presented me with boundless opportunities.

BIOGRAPHY

Aditya P. Goswami was born on 10 June, 1980 in Ahmedabad, India. He received a degree in Electronics and Communication Engineering in May 2001 from the Nirma Institute of Technology, Gujarat, India. He was admitted to the Master's program at North Carolina State University in the Fall of 2001. His interests are in the fields of RF and Mixed Signal Circuit Design.

ACKNOWLEDGEMENTS

I would like to thank everybody who has helped me during my graduate school years and while I was working on my thesis.

I would like to express my sincere gratitude to Dr. Michael B. Steer, my principal advisor, for his support and guidance during my graduate studies, research work and thesis preparation. This thesis would not have been possible without his continuous help.

I would also like to express my sincere appreciation to Dr. Griff Bilbro, and Dr. Douglas Barlage for serving on my M.S. committee and providing me with valuable suggestions. I would like to thank Steve Lipa and Jayesh Nath for helping me out on quite a few occasions.

Most of all, I would like to thank my Mother and Father, who have always stood by me and made it possible for me to pursue graduate studies. My brother Siddharth, has also been a constant source of motivation.

I would like to thank my good friends Shalin and Rachana, who have constantly supported and encouraged me during my Masters.

I would like to thank the Lord for all his blessings.

To all of you, I thank you.

Contents

List of Figures

Chapter 1

Introduction

1.1 Motivation

As clock speeds increase each year, the accurate knowledge of transmission line effects is needed to successfully design and fabricate electronic systems. There are two methods of determining these effects. One, these effects can be predicted by using electromagnetic field theory, also known as analytical modelling, and two, these parameters can be measured experimentally leading to what is known as empirical modelling.

Discontinuities can be described by S-parameters. The Vector Network Analyzer (VNA) is used to measure S-parameters whereby a single frequency sinusoid is applied to the network or Device under Test (DUT) and the reflected and the transmitted energy determine the S-parameters. Measurements by the VNA must have a high degree of accuracy. The are various sources of errors which may occur while doing measurements [8]. Namely:

- *Systematic error*: These are caused by imperfections in the test equipment and test setup. If these errors do not vary over time, they can be characterized through calibration and mathematically removed during the measurement process. Systematic errors encountered in network measurements are related to signal leakage, signal reflections, and frequency dependent loss of fixtures.

- *Random errors*: These errors vary randomly as a function of time. Since they are not predictable, they cannot be removed by calibration. The main contributors to random errors are instrument noise.

- *Drift errors*: These errors occur when a test system's performance changes after a calibration has been performed. They are primarily caused by temperature variation and can be removed by additional calibration. By constructing a test environment with a stable ambient temperature, drift errors can usually be minimized.

The VNA has the ability to remove the systematic errors by using known standards. This process of using known standards to remove systematic errors is known as *vector error correction* or calibration or de-embedding.

Here we discuss a procedure for calibration of the VNA with respect to its application to the measurement of a Printed Circuit Board (PCB) discontinuities. The VNA calibration is done with coaxial standards but the PCB transmission line standards are microstrip and stripline. This requires a fixture for conversion from coaxial to PCB structures. This introduces systematic errors in the measurement. The Through Line (TL) Algorithm determines this error and removes it. Moreover a PCB is manufactured such that without physically altering the board, it is not possible to insert calibration standards. This also enhances the need for Computer Aided Microwave Measurements.

1.2 Thesis Overview

This thesis talks about the Through-Line (TL) calibration process and its implementation using MATLAB as the program NetA (Network Analysis for Microwave Measurements) and LabVIEW. Chapter 2 reviews the basic arguments of symmetry that give rise to the TL calibration process for a non-insertable medium like an Integrated Circuit. Chapter 3 discusses the TL calibration algorithms in detail along with the final de-embedding process for a two port network DUT. Chapter 4 discusses the MATLAB implementation NetA, and flow of these TL algorithms for de-embedding

two-port Microwave Measurements. This chapter also redirects the reader to the Appendix A for further details regarding usage of NetA and its utilities. Chapter 5 discusses the LabVIEW Implementation of NetA. Chapter 6 includes the results obtained using NetA for a test wafer. Finally, we conclude with the summary of the work accomplished and suggestions for future research work in Chapter 7.

This report also includes two appendices. One contains information regarding usage of NetA and each of its utilities. The next is about the guidelines for using the LabVIEW implementation of NetA.

1.3 Original Contributions

The original contributions reported here include:

1. NetA tools for TL calibration and de-embedding of two port networks and results.

2. LabVIEW Implementation of TL calibration and de-embedding.

Chapter 2

Literature Review

2.1 Introduction

There are a number of calibration theories that have been developed over the years. These calibration theories help us determine network parameters and perform network analysis which is fundamental to the understanding and design of all microwave components and systems.

This chapter discusses the fundamentals of a symmetric fixture, advantages of using a symmetric fixture, types of symmetric fixtures and the Through Line calibration process [2] using symmetric fixture. This algorithm is built on the basic TRL calibration [1] process.

2.2 Advantages of using a Symmetric Fixture

In the case of calibration in Network Analysis, there are two considerations. Firstly, there is a need to compensate for imperfections within the calibration standard and secondly, calibration certainty. Calibration certainty is dependent upon prior knowledge of the calibration standards and the choice of a minimum number of required standards. A symmetrical fixture reduces the number of required standards and hence, increases certainty.

2.3 Symmetric Fixture

Let us consider a fixture containing two error networks as shown in Fig. 2.1(a). It has two error networks A and B with respective S-parameters S_{ija} and S_{ijb}. The signal flow graph for one error network is shown in Fig. 2.1(b).

The cascade of the two error networks is the fixture S-parameter set which are designated by $[S_{ijf}]$. A fixture is symmetric if the reflection coefficients are equal and the fixture is reciprocal, i.e. if $S_{11f} = S_{22f}$ and $S_{21f} = S_{12f}$.

Using Mason's non-touching loop rule we see that the fixture S-parameters are related to the individual error network parameters as follows:

$$S_{11f} = S_{11a} + \frac{S_{21a}S_{12a}S_{11b}}{1 - S_{22a}S_{11b}} \tag{2.1}$$

$$S_{22f} = S_{22b} + \frac{S_{22a}S_{21b}S_{12b}}{1 - S_{22a}S_{11b}} \tag{2.2}$$

$$S_{21f} = \frac{S_{21a}S_{21b}}{1 - S_{22a}S_{11b}} \tag{2.3}$$

$$S_{12f} = \frac{S_{12a}S_{12b}}{1 - S_{22a}S_{11b}}. \tag{2.4}$$

Equating for Equations (2.1) and (2.2) and applying reciprocity we find one equation and six complex unknowns:

$$S_{11a}(1 - S_{22a}S_{11b}) + S_{11b}S_{21a}^2 = S_{22b}(1 - S_{22a}S_{11b}) + S_{22a}S_{21b}^2 \tag{2.5}$$

The above equation is the symmetry condition. As symmetry corresponds to the physical nature of the fixture, the symmetric solutions are of two types — first and second order symmetry as discussed below.

2.4 First Order Symmetric Fixture

A first order symmetric fixture, as shown in Figure. 2.2 has identical fixture halves that are asymmetric. The two port error network B is the reverse of A, i.e. $B = A^R$

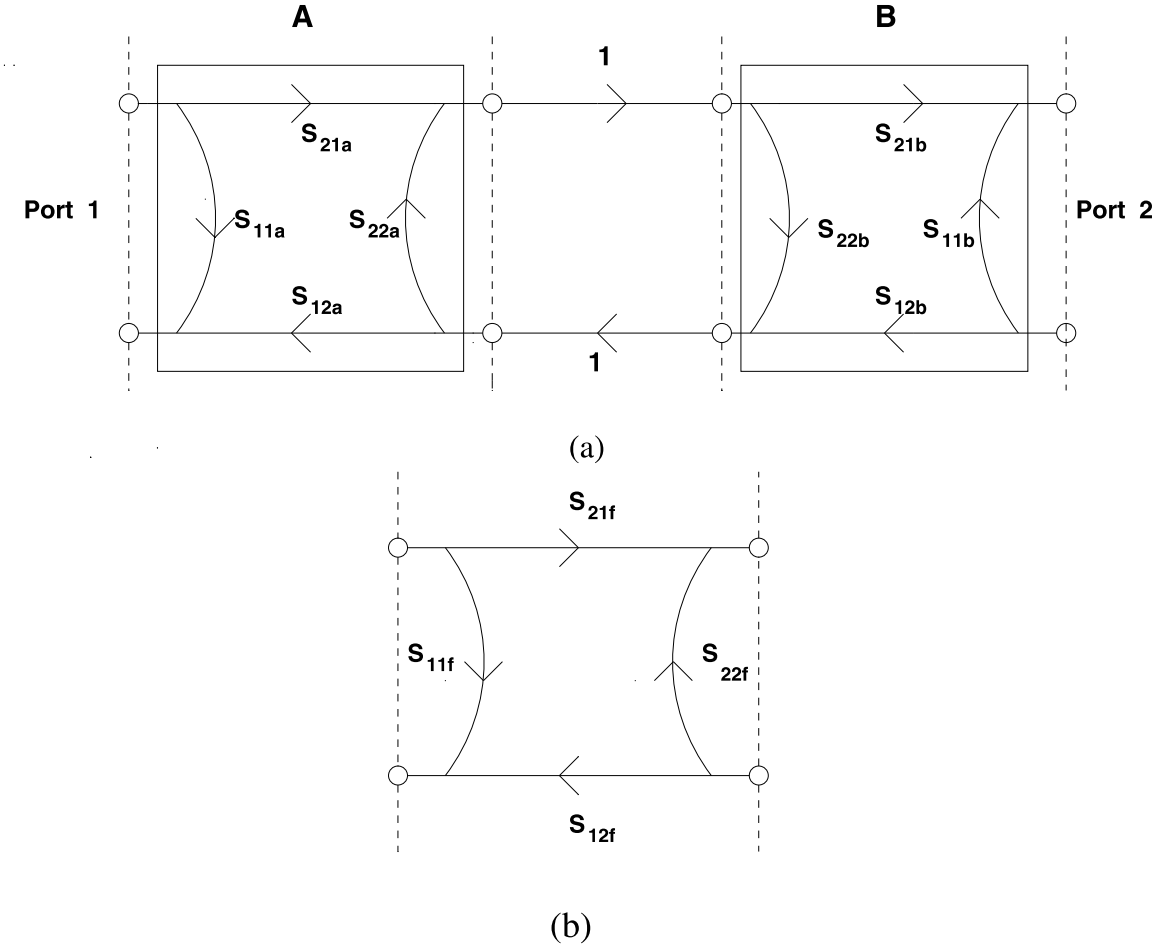


Figure 2.1: Two port model for a fixture: (a) fixture can be described by eight error terms S_{ija} and S_{ijb} where Port 1 and 2 are the pre-calibrated VNA measurement ports; and (b) S_{ijf} are the measured fixture S-parameters.

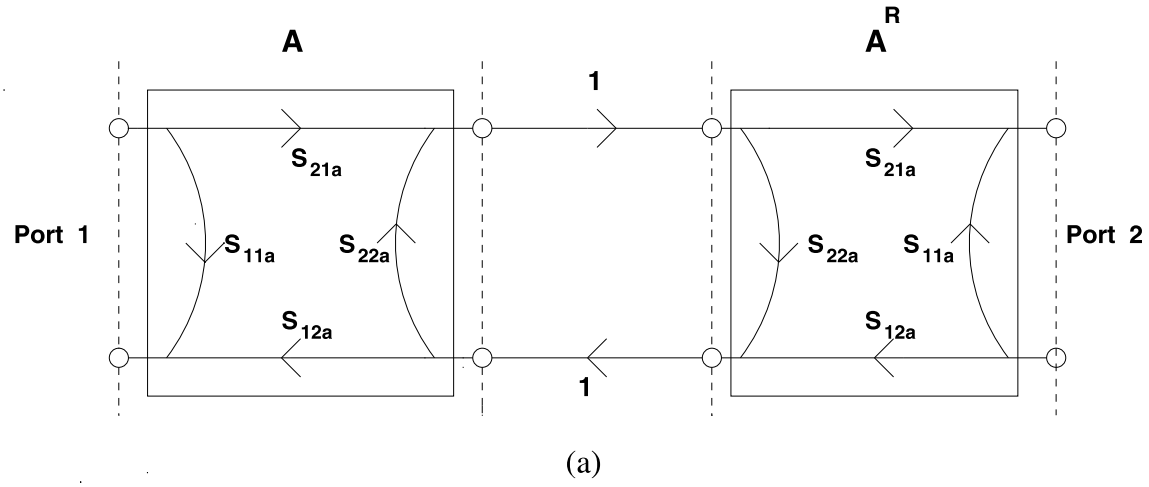


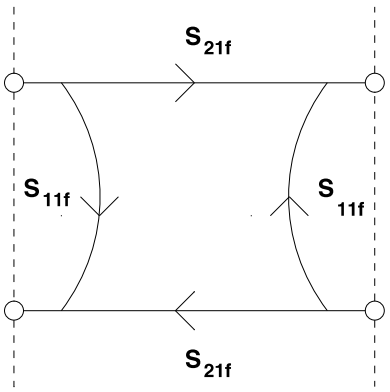
Figure 2.2: Two port error model for a first order symmetric structure: Fixture error networks are described by three S-parameters S_{ija}.

where A and B are network parameters. Hence, $S_{11a} = S_{22b}$, $S_{22a} = S_{11b}$ and $S_{21a}^2 = S_{21b}^2$. Moreover, these conditions satisfy the symmetry condition in Equation (2.5). A coaxial-to-microstrip and microstrip-to- coaxial fixture is an example of first order symmetrical fixturing provided that the two adapters are identical. Most importantly, the signal flow graph also shows that symmetry reduces the number of error terms to three, for the A network only. In other words, symmetry reduces the number of error terms for a two port fixture, which is strictly a function of the manufacturing repeatability with direct application to printed circuit board measurements.

2.5 Second Order Symmetric Fixture

If each of the fixture halves are identical and symmetric as shown in Figure. 2.3, the fixture is a second order symmetric fixture. This implies that error network B is equal to error network A, i.e. the reflection S-parameters, S_{iia} or S_{iib}, are equal to S_{11}, and all transmission S-parameters, S_{ija} or S_{ijb} are equal to S_{21}.

In the next section, we apply the first order symmetry to the microstrip fixture and synthesize the reflection standard from there on.



(b)

Figure 2.3: Two port error model for a first order symmetric structure: S_{11f} and S_{21f} are measured fixture S-parameters.

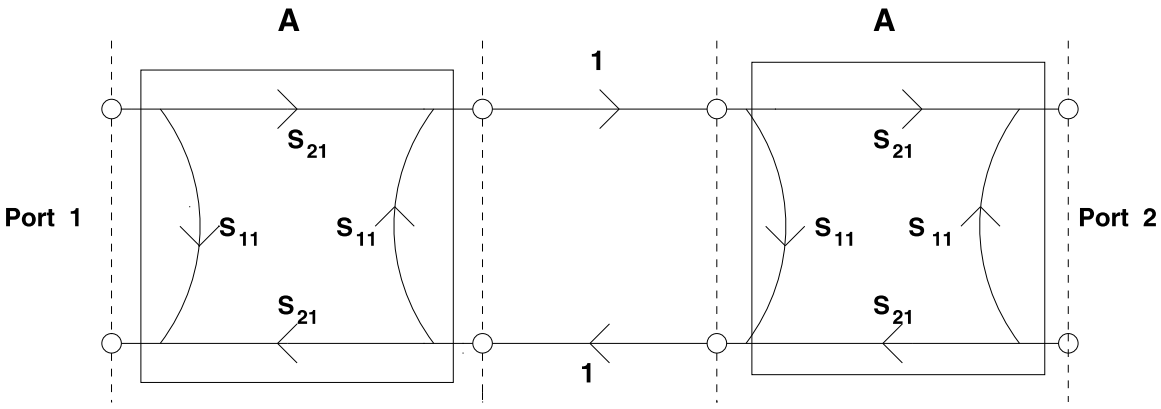


Figure 2.4: Two port error network for a second order symmetric structure. Error network is described by two S-parameters S_{11} and S_{21}.

2.6 Through-Line Using First Order Symmetry

2.6.1 Using Symmetry to Replace the TRL Reflection Standard

For calibration of a planar measurement fixture, e.g. microstrip or stripline, we require distributed standards. The traditionally preferred method for planar fixture de-embedding is Through-Reflect-Line or TRL because the distributed standards can be easily modelled and constructed. A typical microstrip fixture is a first order symmetric fixture. The coaxial-to-microstrip transitions and the microstrip line lengths must be symmetrical. At low microwave frequencies these can be easily achieved and at higher frequencies achieved through careful design.

First order symmetry allows the fixture to be represented by three error terms [3]. Symmetry permits synthesis of the TRL reflection standard, thereby eliminating its need. TRL standards are shown in Figure 2.4 — a through connection, a reflection standard and a reference transmission line of length l. The reflection is arbitrary and is typically a short circuit. As will be shown for the symmetrical through connection we can calculate the reflection coefficient with the microstrip ports terminated in ideal open or short circuits. Either of these synthesized reflection coefficients can replace the TRL reflection standard. Thus, we will refer to this method of using synthesized standards as Through-Line or TL method for calibration.

2.6.2 Synthesize Reflection Standards

The Through Line (TL) standards are the thorough and the line connections, as shown in Figure 2.5. Their results are based on low fixture return loss assumption. We will now look at the derivation of the ideal open and short circuit reflection coefficients as previously presented in [3] and [6].

As seen in the signal flow graphs of Figure 2.2, we consider the through connection S-parameters $S_{11f} = S_{22f}$ and $S_{21f} = S_{12f}$, and the actual parameters of the error network A are δ, α and γ for S_{11a}, $S_{21a} = S_{12a}$ and S_{22a} respectively (refer to Figure. 2.6). The input reflection coefficient with an ideal short circuit placed at the Port 1b

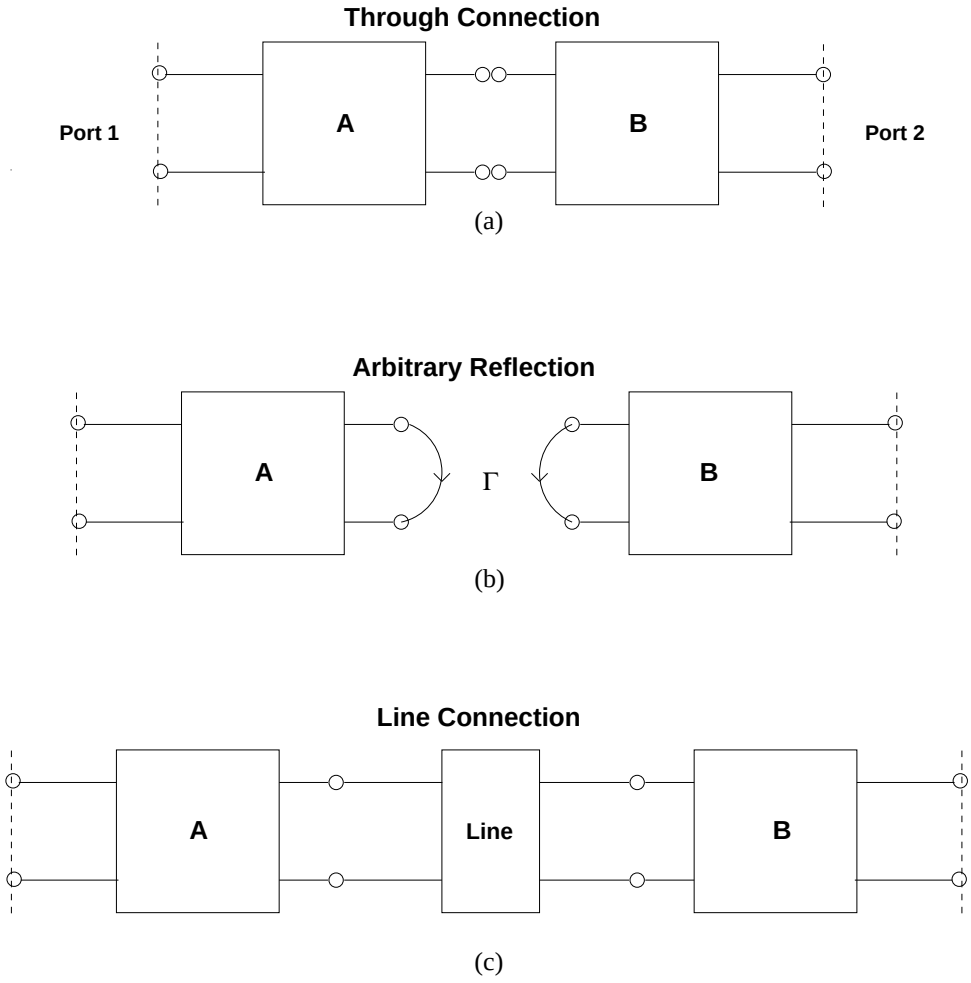


Figure 2.5: Through-Reflect-Line Calibration Standards: (a) Through Standard (b) Reflection Standard and (c) Line Standard.

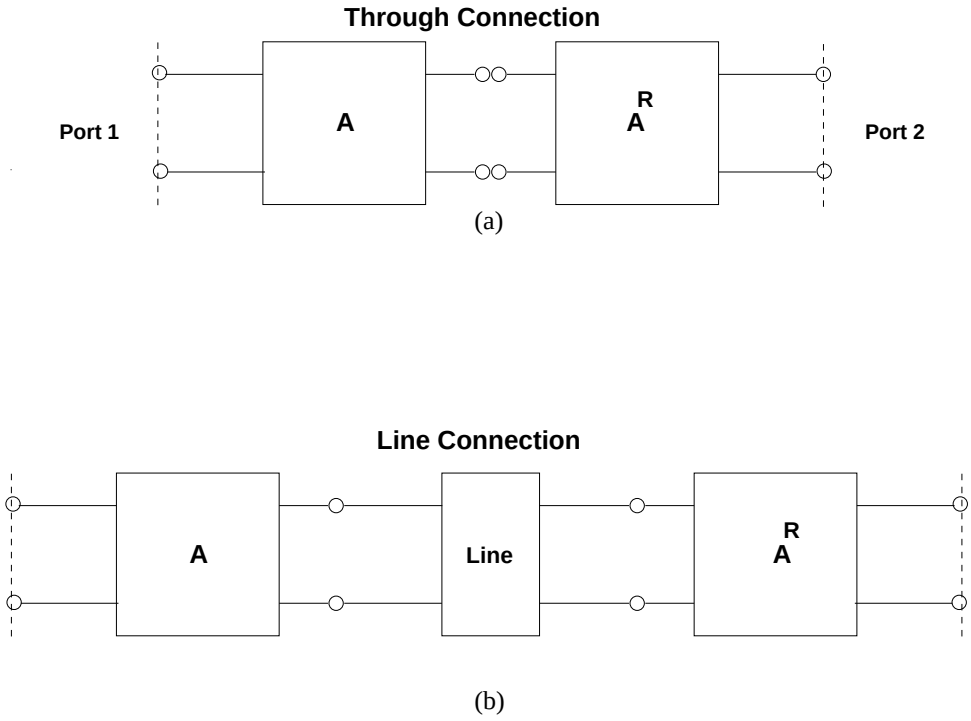


Figure 2.6: Through-Line (TL) Calibration Standards: (a) through connection; and (b) line connection. The TRL reflection standard is synthesized using symmetry.

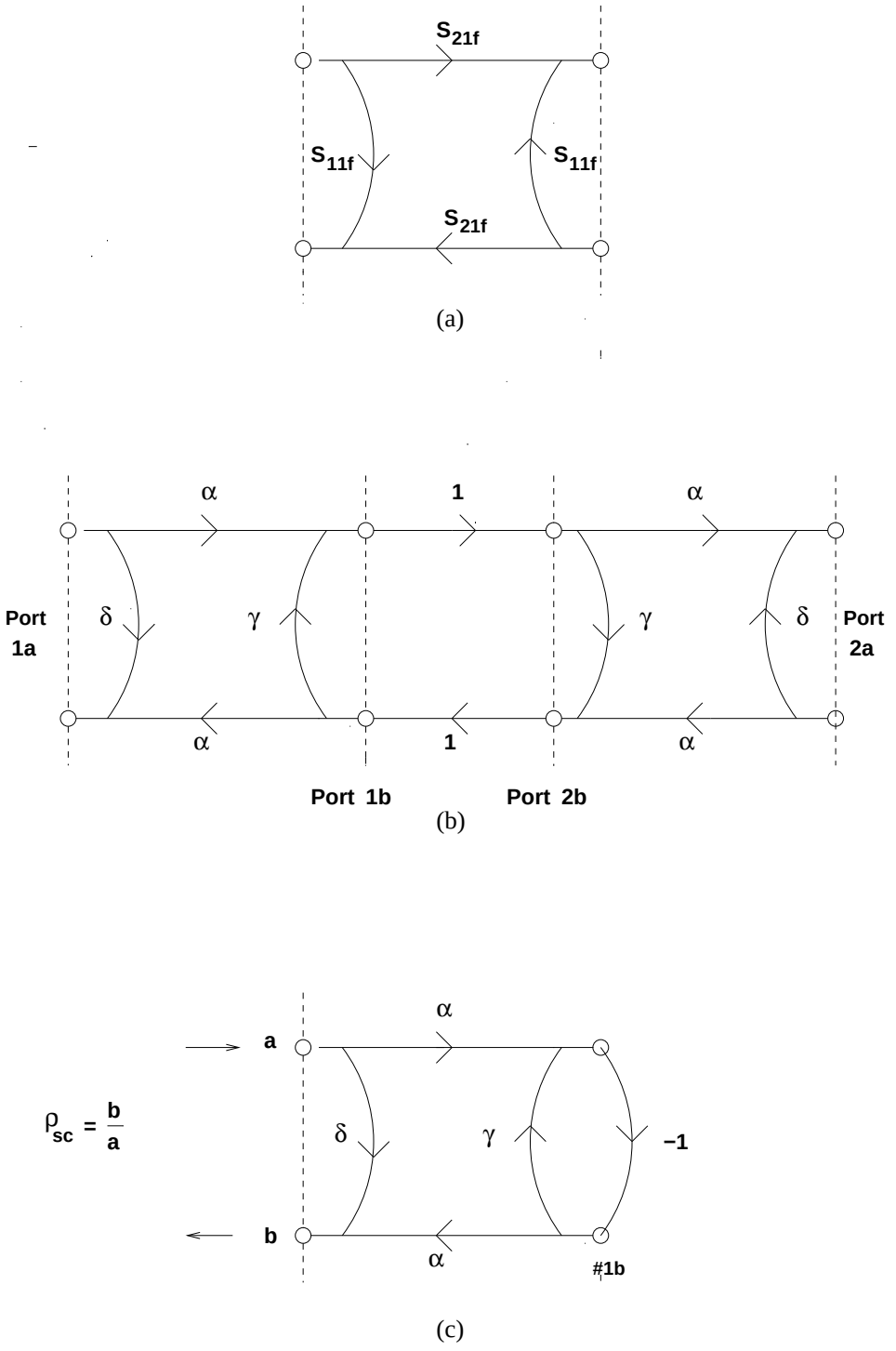


Figure 2.7: TL signal flow graphs: (a) Measured fixture S-parameters; (b) Fixture described by parameters δ, α and γ; and (c) Signal Flow Graph with an ideal short placed at fixture Port 1b.

of A, in Figure 2.6(c), when calculated is

$$\rho_{sc} = \delta - \frac{\alpha^2}{1+\gamma}. \tag{2.6}$$

Previously we used Mason's formula to derive (2.1) to (2.4). For the first order symmetric structure shown in Figure 2.6(b)

$$\begin{aligned} S_{11f} &= \delta + \frac{\alpha^2\gamma}{1-\gamma^2} \\ S_{21f} &= \frac{\alpha^2}{1-\gamma^2}. \end{aligned} \tag{2.7}$$

Similarly, for an open circuit placed at the Port 1b of A gives the result

$$\rho_{oc} = \delta + \frac{\alpha^2}{1-\gamma}. \tag{2.8}$$

From Equations (2.6)–(2.9) the short circuit reflection coefficient and the open circuit reflection coefficients can be expressed as a function of the measured fixture S-parameters as

$$\begin{aligned} \rho_{sc} &= S_{11f} - S_{21f} \\ \rho_{oc} &= S_{11f} + S_{21f} \end{aligned} \tag{2.9}$$

The above results are used in the later chapters while explaining the Through Line Calibration Algorithm in detail. Compared to TRL, TL reduces the number of standards needed and requires fewer connections to the microstrip ports and thus improves the repeatability of this connection. Moreover, ρ_{sc} and ρ_{oc} can be derived for any fixture with a first or second order symmetry. As these reflection coefficients are derived mathematically, it is possible to insert an ideal open or an ideal short circuit within the non-insertable medium like a dielectric loaded waveguide too.

2.7 Summary

In this chapter it is seen that in the TL calibration the reflection standard of the TRL calibration is synthesized from the through measurement by using a perfect short or

open. Thus, by using symmetric arguments, it is possible to reduce the number of required calibration standards. Chpater 3 discusses the TL algorithms as well as the procedural flow for the calibration and de-embedding process.

Chapter 3

Through-Line Algorithms

3.1 Introduction

This chapter explains the algorithms used in the *Through-Line* calibration procedure. These algorithms have been implemented as MATLAB code. This implementation enables raw S-parameters of the *through* and the *line* calibration standards (refer to Figure 3.1) to be read; applies the various calibration procedures; and hence obtains the de-embedded result. Moreover it has various utility routines that can be used for data analysis.

These calibration routines perform four main functions to obtain the de-embedded result. Namely:

- propagation constant determination
- Z_c (characteristic impedance) determination
- error matrix determination
- de-embedding algorithm

Each of the above algorithms will be discussed in detail in the following sections.

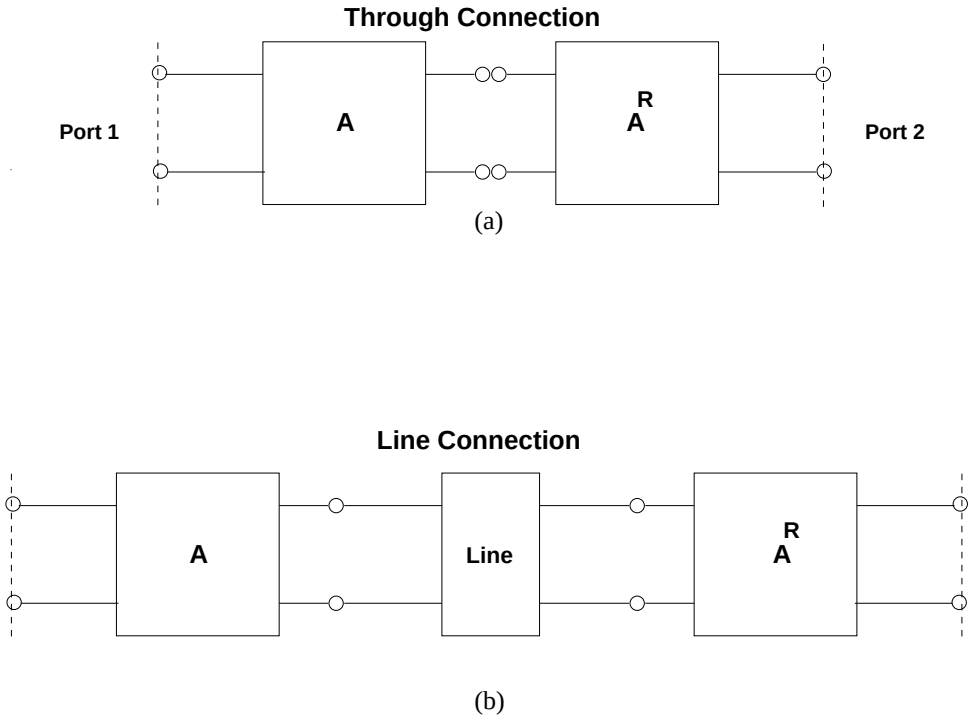


Figure 3.1: Through-Line (TL) Calibration Standards: (a) through connection; and (b) line connection. The TRL reflection standard is synthesized using symmetry.

3.2 Theory

The first step in the algorithm is to read in the raw S-parameters of the *through* and the *line* calibration structures. These S-parameters are then converted into T-parameters or ABCD parameters. The equations used for this conversion are

$$T_{11} = ((1+s_{11})\cdot(1-s_{22})+s_{12}\cdot s_{21})/2\cdot s_{21} \quad (3.1)$$

$$T_{12} = z0\cdot((1+s_{11})\cdot(1+s_{22})-s_{12}\cdot s_{21})/2\cdot s_{21} \quad (3.2)$$

$$T_{21} = ((1-s_{11})\cdot(1+s_{22})-s_{12}\cdot s_{21})/2\cdot z0\cdot s_{21} \quad (3.3)$$

$$T_{22} = ((1-s_{11})\cdot(1+s_{22})+s_{12}\cdot s_{21})/2\cdot s_{21} \quad (3.4)$$

3.2.1 Propagation Constant Determination

This routine requires the T-parameters of the through and line structures and difference in lengths of the through and the line structures. Mondal and Chen [5] showed that the TRL Algorithm can be used to determine the propagation constant of the TRL Line Standard. This is given by

$$\gamma = \ln(A \pm \frac{\sqrt{A^2-4}}{2})\frac{1}{(l_1-l_2)} \quad (3.5)$$

where

l_1 = length of the line calibration structure.

l_2 = length of the through calibration structure.

and A in (3.5) is given by

$$A = T_{11t}T_{22l}+T_{11l}T_{22t}-T_{21t}T_{12l}-T_{12t}T_{21l} \quad (3.6)$$

where T_{ijt} and T_{ijl} in (3.6) are the chain scattering parameters for the through and line calibration standards respectively.

One of the most important parts of the algorithm is deciding the phase of the propagation constant. The phase depends upon the difference in the lengths of the through and the line calibration standards. However, if the electrical length of the transmission line is $\lambda/2$, measurement errors are observed. Reliable propagation

constant values can be extracted between 20 and 160 degree phase differences. So, the lengths should only be such that this problem is avoided over the entire measurement frequency range. This appears to be particular to the TRL procedure. However, generally good results are obtained using the TL calibration procedure even when the phase is not in this range.

3.2.2 Characteristic Impedance Determination

An inherent assumption of the TRL algorithm is that the characteristic impedance of the line calibration standard is either equal to the measurement system impedance or can be precisely determined. A technique was proposed in [4] in which the characteristic impedance is calculated using the Enhanced TRL Technique (ETRL) technique. This is an enhancement to the TRL algorithm. If the medium used is a dispersive medium like the microstrip line, then the frequency dependence of Z_c(characteristic impedance) needs to be explored.

ETRL determines the complex characteristic impedance using the free space capacitance C_o, of the line and the propagation constant, $\gamma = \alpha + j\beta$, determined in the standard TRL Algorithm. This impedance is then used in the TRL algorithm to obtain characteristic impedances of any arbitrary TEM transmission lines. In the next subsection the calculation of the complex characteristic impedance of a microstrip transmission line is shown.

3.2.3 Calculation of Complex Characteristic Impedance

Many lines, such as microstrip lines, support a quasi-TEM mode. However, the TL and the TRL algorithm require the equivalent TEM mode characteristic impedance as the reference plane is perpendicular to the direction of propagation and it is to this plane that the impedances are referred. For a uniform TEM transmission lines the characteristic impedance,

$$Z_c = \sqrt{\frac{L}{C}} = v_p L = \frac{1}{v_p C} \tag{3.7}$$

and

$$v_p = \frac{1}{\sqrt{LC}} \tag{3.8}$$

where,

v_p is the phase velocity.

L is the series inductance per unit length, and

C is the shunt capacitance per unit length. Now,

$$c = \frac{1}{\sqrt{LC_o}} \tag{3.9}$$

and so,

$$Z_o = \sqrt{\frac{L}{C_o}} = cL = \frac{1}{cC_o} \tag{3.10}$$

where,

c is the velocity of light in free-space.

C_o is the free-space capacitance.

Hence,

$$L = \frac{1}{c^2 C_o}. \tag{3.11}$$

In the dielectric medium the phase velocity v_p becomes

$$v_p = \frac{c}{\sqrt{\epsilon_r}} \tag{3.12}$$

or

$$v_p = \frac{c}{\sqrt{\frac{C}{C_o}}} \tag{3.13}$$

where,

ϵ_r is the relative dielectric constant.

Now, from (3.7) and (3.11), we have several expressions for the characteristic impedance:

$$Z_c = \sqrt{\frac{1}{c^2 C_o C}} \tag{3.14}$$

$$Z_c = \frac{1}{c\sqrt{CC_o}} \tag{3.15}$$

$$Z_c = \frac{1}{cC_o\sqrt{\frac{C}{C_o}}} \tag{3.16}$$

$$Z_c = \frac{Z_o}{\sqrt{\frac{C}{C_o}}} \tag{3.17}$$

and

$$Z_c = \frac{Z_o}{\sqrt{\epsilon_r}}. \tag{3.18}$$

Moreover, the propagation constant

$$\gamma = jw\sqrt{\mu\epsilon} \tag{3.19}$$

$$\gamma = jw\sqrt{\mu_o\mu_r\epsilon_o\epsilon_r} \tag{3.20}$$

$$\gamma = \frac{jw}{c}\sqrt{\mu_r\epsilon_r} \tag{3.21}$$

and

$$Z_c = Z_o\sqrt{\frac{\mu_r}{\epsilon r}}. \tag{3.22}$$

Hence from (3.10), (3.21) and (3.22),

$$Z_c = \frac{-j\gamma}{C_o\omega\epsilon_r} \tag{3.23}$$

The Through Line calibration algorithm like the TRL procedure requires a reflectionless line. So, each calibration measurement is transformed from the system reference impedance to Z_c. The application of the TRL algorithm is then used to determine the S-parameters of the error network which is referenced to Z_c. Now, the S-parameters of the error network are referred back to the system reference impedance which is usually 50 Ω.

3.2.4 Error Network Determination

As mentioned earlier, the error network is determined using the TRL algorithm. The procedure of finding the S-parameters of error network A (refer to Figure 3.2(a)) is explained below. Error network determination begins by converting the S-parameters of the through and the line calibration standards into wave cascading parameters or a wave cascading matrix by the following formula proposed by Engen and Hoer [1].

$$\begin{bmatrix} b_1 \\ a_1 \end{bmatrix} = \frac{1}{s_{21}} \begin{bmatrix} -\Delta & s_{11} \\ -s_{22} & 1 \end{bmatrix} \begin{bmatrix} a_2 \\ b_2 \end{bmatrix} \tag{3.24}$$

Hence,

$$\begin{bmatrix} b_1 \\ a_1 \end{bmatrix} = \mathbf{R} \begin{bmatrix} a_2 \\ b_2 \end{bmatrix} \tag{3.25}$$

where $\Delta = s_{11}s_{22} - s_{12}s_{21}$ and $\mathbf{R}$ is the *wave cascading matrix*. Moreover an important property of the $\mathbf{R}$ matrix is that the cascade of two or more two-ports is merely the product of the individual R matrices. Let us consider $\mathbf{R_a}$ and $\mathbf{R_b}$ to be the $\mathbf{R}$ matrices of the error networks A and B as shown in Figure 3.1. Now, for the through connection:

$$\mathbf{R_t} = \mathbf{R_a} \cdot \mathbf{R_b} \tag{3.26}$$

where $\mathbf{R_t}$ is the $\mathbf{R}$ matrix for the through connection. Similarly for the line connection, $\mathbf{R_d}$

$$\mathbf{R_d} = \mathbf{R_a} \cdot \mathbf{R_l} \cdot \mathbf{R_b} \tag{3.27}$$

where $\mathbf{R_l}$ represents the line that has been inserted. Now, from (3.25) and (3.26),

$$\mathbf{R_d} = \mathbf{R_a} \cdot \mathbf{R_l} \cdot \mathbf{R_t} \cdot \mathbf{R_a^{-1}} \tag{3.28}$$

$$\mathbf{R_d} \cdot \mathbf{R_t^{-1}} \cdot \mathbf{R_a} = \mathbf{R_a} \cdot \mathbf{R_l} \tag{3.29}$$

Hence

$$T\mathbf{R_a} = \mathbf{R_a R_l} \tag{3.30}$$

where

$$T = \mathbf{R_d R_t^{-1}} \tag{3.31}$$

If we assume a non-reflecting line with propagation constant γ and length l, then

$$R_l = \begin{bmatrix} e^{-\gamma l} & 0 \\ 0 & e^{\gamma l} \end{bmatrix} \tag{3.32}$$

Since S_{11} =0 and $S_{22} = 0$, using (3.30) and (3.32),

$$\begin{bmatrix} t_{11} & t_{12} \\ t_{21} & t_{22} \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} \\ r_{21} & r_{22} \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} \\ r_{21} & r_{22} \end{bmatrix} \begin{bmatrix} e^{-\gamma l} & 0 \\ 0 & e^{\gamma l} \end{bmatrix} \tag{3.33}$$

where $[r_{ij}]$ is the wave cascading matrix R_a of error network A and $[t_{ij}]$ is the wave cascading matrix of the line. Hence,

$$t_{11}r_{11} + t_{12}r_{21} = r_{11}e^{-\gamma l} \tag{3.34}$$

$$t_{21}r_{11} + t_{22}r_{21} = r_{21}e^{-\gamma l} \tag{3.35}$$

$$t_{11}r_{12} + t_{12}r_{22} = r_{12}e^{\gamma l} \tag{3.36}$$

$$t_{21}r_{12} + t_{22}r_{22} = r_{22}e^{\gamma l} \tag{3.37}$$

Dividing (3.34) by (3.35), and (3.36) by (3.37):

$$t_{21}\left(\frac{r_{11}}{r_{21}}\right)^2 + (t_{22} - t_{11})\left(\frac{r_{11}}{r_{21}}\right)^2 - t_{12} = 0 \tag{3.38}$$

$$t_{21}\left(\frac{r_{12}}{r_{22}}\right)^2 + (t_{22} - t_{11})\left(\frac{r_{12}}{r_{22}}\right)^2 - t_{12} = 0 \tag{3.39}$$

It is assumed that $a = r_{11}/r_{22}$, $b = r_{12}/r_{22}$, $c = r_{21}/r_{22}$ and $ac = r_{11}/r_{21}$. Hence, $ac = a/c$. By solving (3.38) and (3.39), the values for the roots ac and b are obtained. Now from Equation (38) of Engen and Hoer [1],

$$a = \frac{w_1 - b}{\Gamma_l(1 - w\frac{c}{a})} \tag{3.40}$$

Here $\Gamma_l = -1$ as a perfect short is used and $w_1 = \rho_{sc}$ (see Chapter 2). Hence,

$$a = \frac{\rho_{sc} - b}{-1(1 - \rho_{sc}\frac{c}{a})} \tag{3.41}$$

As $ac = a/c$:

$$a = \frac{\rho_{sc} - b}{-1(1 - \rho_{sc}\frac{1}{ac})} \tag{3.42}$$

The main objective here is to find the S-parameters of the error boxes A and B. Using (3.26) and assuming that α, β, γ and ρ are the wave cascading parameters for error box B and d, e, f and g are the wave cascading parameters for the Through connection,

$$r_{22}\rho_{22}\begin{bmatrix} a & b \\ c & 1 \end{bmatrix}\begin{bmatrix} \alpha & \beta \\ \gamma & 1 \end{bmatrix} = g\begin{bmatrix} d & e \\ f & 1 \end{bmatrix} \tag{3.43}$$

As error box A and B are symmetrical, it can be said that $r_{22} = \rho_{22}, \alpha = a, \beta = -c$ and $\gamma = -b$. Substituting these values in (3.43) yields

$$r_{22}^2\begin{bmatrix} a & b \\ c & 1 \end{bmatrix}\begin{bmatrix} a & -c \\ -b & 1 \end{bmatrix} = g\begin{bmatrix} d & e \\ f & 1 \end{bmatrix} \tag{3.44}$$

$$r_{22}^2\begin{bmatrix} a^2 - b^2 & b - ac \\ ac - b & 1 - c^2 \end{bmatrix} = g\begin{bmatrix} d & e \\ f & 1 \end{bmatrix} \tag{3.45}$$

Let us assume that $(a^2 - b^2) = x$, $(b - ac) = y$, $(ac - b) = w$ and $(1 - c^2) = z$ and solving (3.45) further gives:

$$r_{22}^2\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \frac{g}{(xz - yw)}\begin{bmatrix} d & e \\ f & 1 \end{bmatrix}\begin{bmatrix} z & w \\ y & x \end{bmatrix} \tag{3.46}$$

Substituting back the values for x, y, w and z yields:

$$r_{22}^2 \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \frac{g}{(a^2 - 2abc + b^2)} \begin{bmatrix} d & e \\ f & 1 \end{bmatrix} \begin{bmatrix} 1 - c^2 & ac - b \\ b - ac & a^2 - b^2 \end{bmatrix} \tag{3.47}$$

$$r_{22}^2 \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \frac{g}{(a - bc)^2} \begin{bmatrix} d & e \\ f & 1 \end{bmatrix} \begin{bmatrix} 1 - c^2 & ac - b \\ b - ac & a^2 - b^2 \end{bmatrix} \tag{3.48}$$

$$r_{22}^2 \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \frac{g}{(a - bc)^2} \begin{bmatrix} d(1 - c^2) + e(b - ac) & d(ac - b) + e(a^2 - b^2) \\ f(1 - c^2) + (b - ac) & f(ac - b) + (a^2 - b^2) \end{bmatrix} \tag{3.49}$$

Equating both sides of (3.49) four equations are obtained:

$$r_{22}^2 = d(1 - c^2) + e(b - ac) \tag{3.50}$$

$$r_{22}^2 = f(ac - b) + (a^2 - b^2) \tag{3.51}$$

$$d(ac - b) = -e(a^2 - b^2) \tag{3.52}$$

$$f(1 - c^2) = (ac - b). \tag{3.53}$$

Substituting the value of d from (3.52) in (3.50),

$$r_{22}^2 = \frac{g}{(a - bc)^2}(-e\frac{(a^2 - b^2)(1 - c^2)}{(ac - b)} - e(ac - b)) \tag{3.54}$$

$$r_{22}^2 = \frac{-ge}{(a - bc)^2}\frac{(a^2 - b^2)(1 - c^2) + (ac - b)^2}{(ac - b)} \tag{3.55}$$

$$r_{22}^2 = \frac{-ge}{(a - bc)^2}\frac{(a^2 - b^2 - a^2c^2 + b^2c^2 + a^2c^2 + b^2 - 2abc)}{(ac - b)} \tag{3.56}$$

$$r_{22}^2 = \frac{-ge(a - bc)^2}{(a - bc)^2(ac - b)} \tag{3.57}$$

and finally

$$r_{22} = \sqrt{\frac{-ge}{ac-b}}. \tag{3.58}$$

As values of a, b, c and r_{22} have been obtained, the wave cascading matrix for error network A can now be built by finding,

$$r_{11} = r_{22} \times a \tag{3.59}$$

$$r_{12} = r_{22} \times b \tag{3.60}$$

$$r_{21} = r_{22} \times c \tag{3.61}$$

Hence, following conversion of these wave cascading parameters to S-parameters, the S-parameters of error box A are obtained.

3.2.5 De-embedding Algorithm

As the error boxes A and B are assumed to be symmetrical, the S-parameters for error box B are obtained by reversing the S-parameters of error box A. Thus, $S_{error,A}$ and $S_{error,B}$ are obtained. We now, put the two-port network whose de-embedded S-parameters S_{dut} are to be obtained and measure its S-parameters (which are inclusive of the assumed error boxes) and are given by S_{meas} (Refer Figure 3.2). Thus,

$$S_{meas} = S_{error,A} \cdot S_{dut} \cdot S_{error,B} \tag{3.62}$$

$$S_{dut} = S_{error,A}^{-1} \cdot S_{meas} \cdot S_{error,B}^{-1} \tag{3.63}$$

S_{dut} here is the de-embedded S-parameters for the measured two port network. Here the above error matrices and measured parameters are first converted into T-parameters, de-embedded results are now obtained in terms of T-parameters and then converted back to S-parameters.

3.3 Summary

In Chapter 3 we discussed the TL calibration algorithms in detail along with the final de-embedding process for a two port network DUT. The next chapter discusses

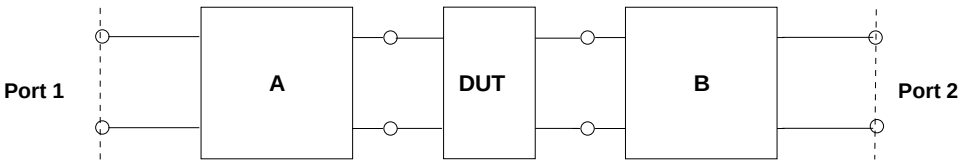


Figure 3.2: Final De-embedding of a Two-port Device Under Test. S_{meas} here are the measured S-parameters for the cascade of error networks A,B and the DUT

the MATLAB implementation and flow of these TL algorithms for de-embedding two-port microwave measurements.

Chapter 4

NetA

4.1 Introduction to NetA

NetA is a Computer Program developed to enhance the capability of Computer Aided Microwave Measurements. NetA enables engineers to perform the *Through-Line* (TL) calibration and de-embedding procedure for two port networks. NetA is a tool which is used for processing S-parameter data and its prime use is to implement the Through-Line de-embedding algorithm. NetA also has a number of utilities that help in the Microwave Measurement analysis for two port networks.

NetA has been written in MATLAB, which is a programming language widely used by engineers. It has been developed in a very structured format and is also upgradable. Moreover, other de-embedding techniques can be included as MATLAB subroutines to enhance its capability, while utilizing the same MATLAB routines for data analysis. Each routine can be called through the command line in MATLAB. Moreover, these MATLAB routines have been used to develop a LabVIEW implementation which shall be explained in the following chapter.

4.2 NetA Algorithm Flow

As explained earlier the core de-embedding algorithm has been divided into four major parts. Namely,

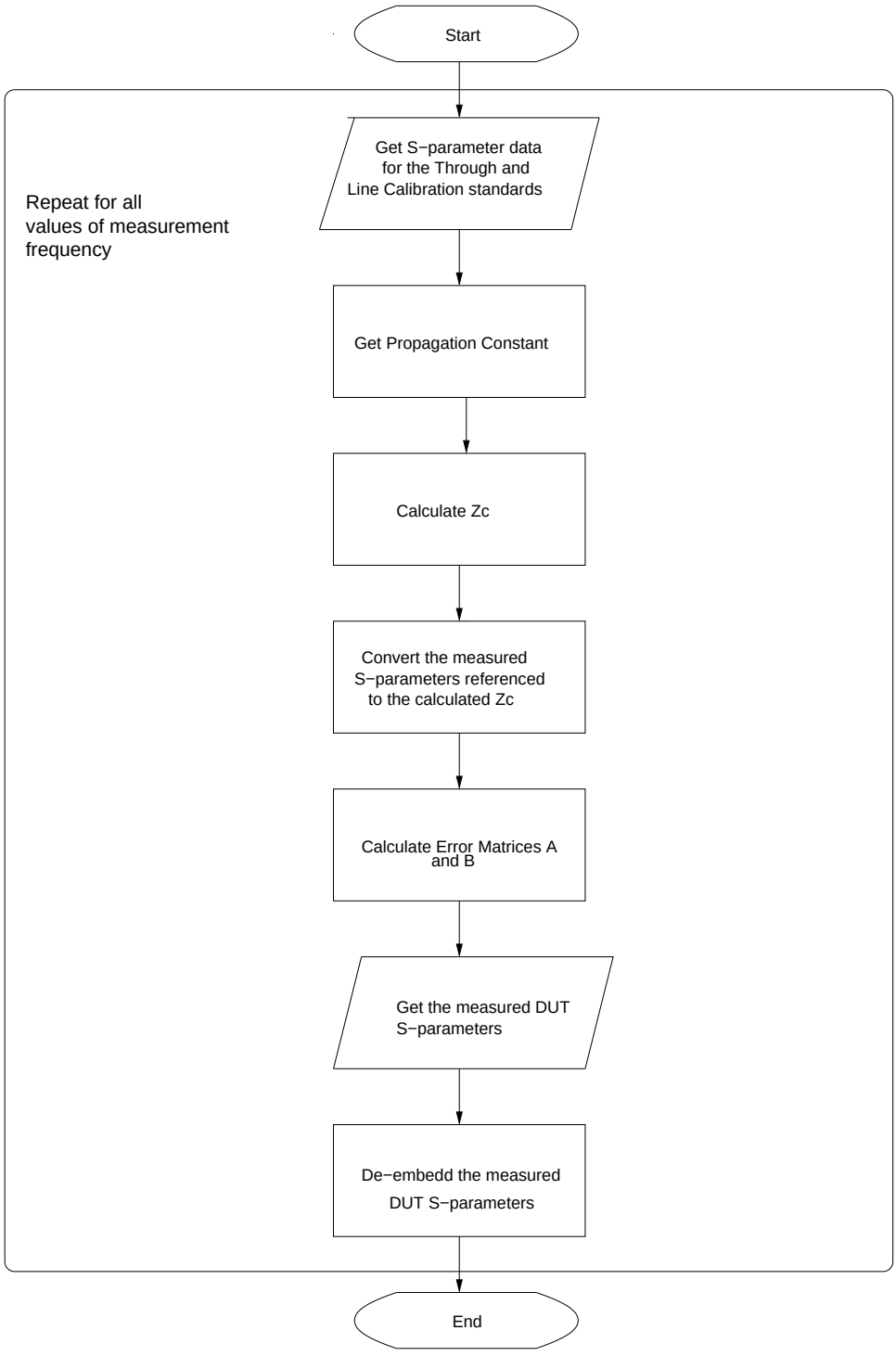


Figure 4.1: Flow Chart for the NetA Algorithm.

- Propagation Constant Determination
- Z_c (characteristic impedance) Determination
- Error Matrix Determination
- De-embedding Algorithm

Each of the above mentioned algorithms was discussed in detail in Chapter 3. Figure 4.1 shows the flow chart for the Through-Line de-embedding algorithm.

4.3 Utilities available in NetA

In MATLAB, for every command executed the resultant arrays or matrices are displayed with their name, dimension and contents. So, any of the utility subroutines can be individually executed by passing the required data as input arguments. Error display occurs in case there is a violation. To understand the semantics of each command the user is directed to Appendix A. Below is a list of the MATLAB utilities and a brief discussion of their function. These utilities can be used to implement the TL and the TRL calibration procedures.

	Function	Purpose
1.	GET_MAG_ANG	Input Data
2.	TL	Error Matrix Calculation
3.	TL_CALIB	Through Line Calibration Routine
4.	TSL	Two port De-embedding
5.	CAL_GAMMA	Calculates the Propagation Constant
6.	STOR	Converts S-parameters to R-parameters
7.	RTOS	Converts R-parameters to S-parameters
8.	PTOR	Converts Polar to Rectangular
9.	STOZ	Converts S-parameters to Z-parameters
10.	SREVERSE	Reverses the input S-parameter network
11.	ZTOS	Converts Z-parameters to S-parameters
12.	STOH	Converts S-parameters to H-parameters
13.	HTOS	Converts H-parameters to S-parameters
14.	STOY	Converts S-parameters to Y-parameters
15.	YTOS	Converts Y-parameters to S-parameters
16.	STOT	Converts S-parameters to T-parameters
17.	TTOS	Converts T-parameters to S-parameters
18.	MAGPHASE	Converts to Magnitude–Angle(degrees) format

4.4 Summary

MATLAB can be used to directly implement the TL calibration and de-embedding process. Various utility routines provided, help for further two port network analysis. More information about the command line syntax and parameters required for implementation and usage of these routines has been provided in Appendix A. The next chapter discusses the LabVIEW Implementation of NetA.

Chapter 5

LabVIEW Implementation

5.1 Brief Introduction to LabVIEW

LabVIEW is both an instrument control program and program development application. LabVIEW uses a graphical programming language, G, to create programs in a block diagram form.

LabVIEW is a general purpose programming tool but it also includes libraries of functions and development tools designed specifically for data acquisition and instrument control. LabVIEW programs are called *Virtual Instruments* (VIs) because their appearance and operation can imitate actual instruments.

5.2 Basic VI Structure

A VI consists of an interactive user interface, a data flow diagram that serves as a source code, and icon connections that allow the VI to be called from a higher level VI. Vis are structured as follows:

- The interactive user interface of a VI is called a *frontpanel*, because it simulates the panel of a physical instrument. The front panel can contain knobs, push buttons, graphs, controls and indicators.

- The VI receives instructions from the block diagram, which you construct. Ba-

sically, the block diagram is a pictorial solution to a programming problem. It is the source code of the VI.

- VIs are hierarchical and modular. You can use them as a top-level program, or as a subVI, which is a subprogram within a program. VIs can pass on data to subVIs.

Hence, with the above features LabVIEW promotes and adheres to the concept of modular programming. One can divide an application into a series of tasks and then subdivide it to decrease the complexity of the required application. Hence, a VI can be built to accomplish each subtask and then by combining all these subVIs on another block diagram a larger task can be accomplished. Finally, the top-level VI contains a collection of subVIs that represent functions or tasks.

The above explanation aids us in understanding the LabVIEW implementation of NetA. Moreover, two major practical advantages of integrating NetA within LabVIEW are:

- Usability of NetA is improved as it is used in conjunction with LabVIEW.
- LabVIEW can be interfaced real-time with the Network Analyzer, hence, saving a lot of measurement and data analysis time.

5.3 LabVIEW Implementation of NetA

The LabVIEW implementation has been divided into four major blocks. These four blocks perform the following functions respectively.

1. The first block reads in the input S-parameter data of the Through and the Line calibration structures for all measurement frequencies.
2. The second block performs the calculation of the propagation constant γ, complex characteristic impedance Z_c and error network parameters for all measurement frequencies.

3. The third block reads in the S-parameter data for the DUT that is to be de-embedded.

4. The fourth and final block performs the de-embedding on the measured embedded DUT S-parameters.

Each of these above blocks is a subVI. These subVIs are then combined on a separate VI block diagram to implement the Through Line De-embedding procedure. The next four sub-sections explain the LabVIEW Implementation of each of the above mentioned four blocks in detail.

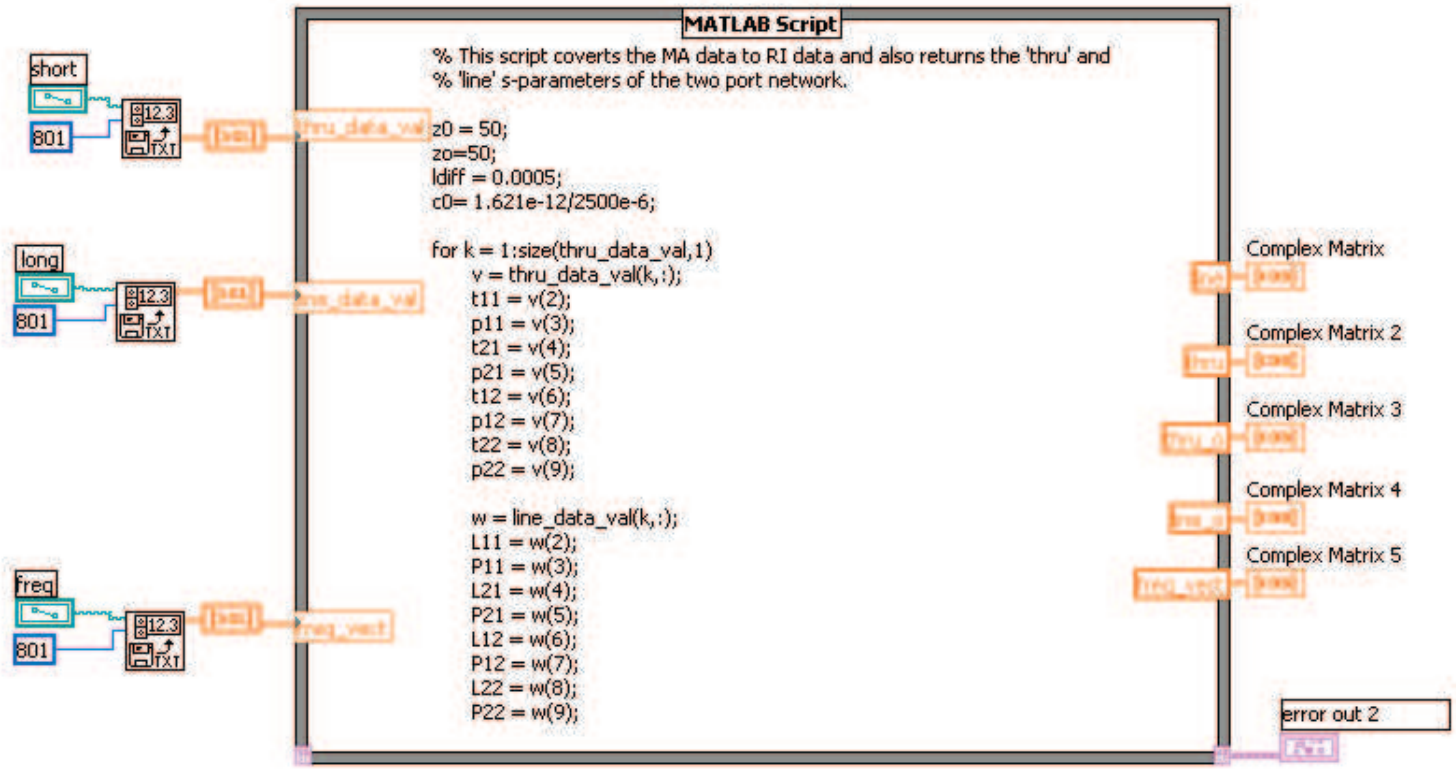


Figure 5.1: Block Diagram for the VI which reads in Through and Line Data.

5.3.1 First Block

The screenshot in Figure. 5.1 is of the first VI. This VI performs the function of reading in the S-parameters for the Through and the Line calibration standards. This data is read from text files and then fed in as an input to the Matlab Script arithmetic structure, as can be seen in the Figure 5.1. The Matlab Script structure is an in-built function within LabVIEW.

Within this Matlab Script structure, the Matlab code is written. Some changes are made in the basic NetA Code to successfully incorporate it within the LabVIEW environment. The input data in the text file is in the magnitude-angle format. This data is first converted into real-imaginary format and then converted to its equivalent

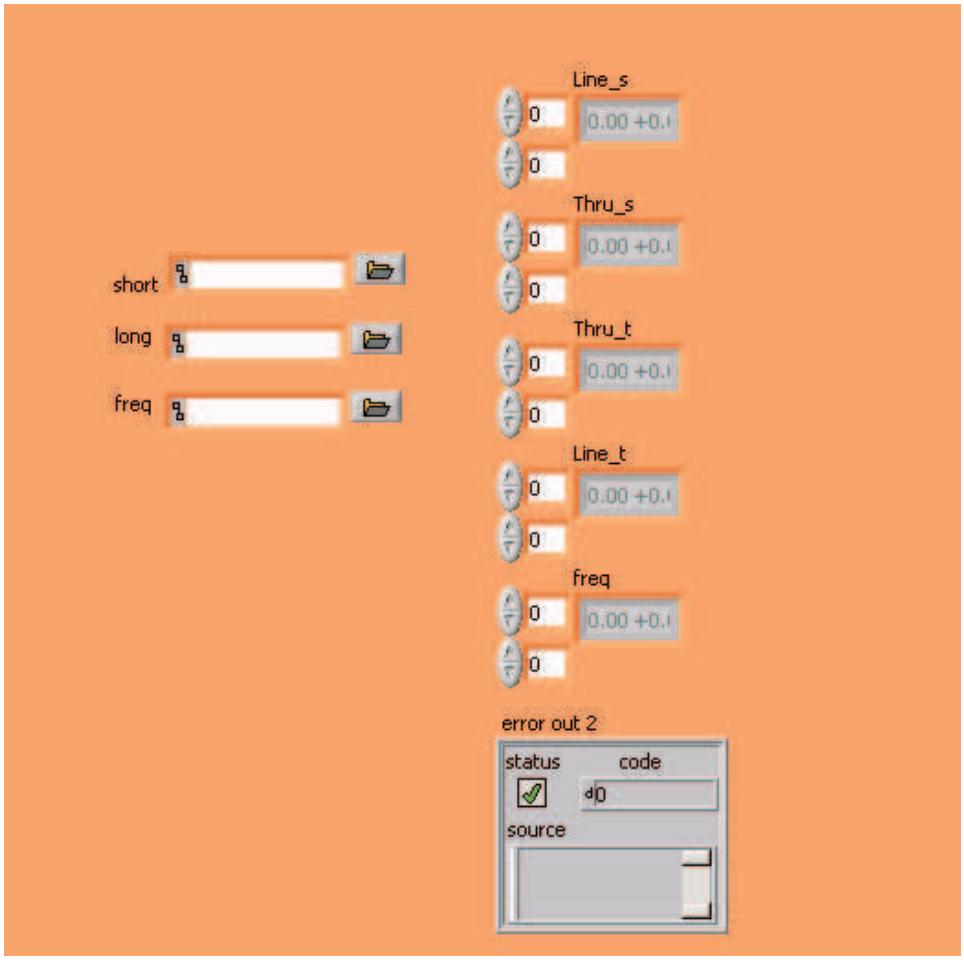


Figure 5.2: Front Panel for the Block Diagram in Figure 5.1

T-parameters for both the calibration standards, for all measurement frequencies and then fed to the output of the VI's block diagram. Each input to the block diagram is a *control* and each output of the block diagram is an *indicator*. Each control and indicator is displayed on the front panel of this VI. There is also an error-out signal associated with each VI that displays the error, if any, during simulation. The resultant front panel for the VI along with all controls and indicators is shown in Figure 5.2. We also prepare an *icon-connector pane* for this VI so that it can be used as a subVI at a higher level of abstraction during implementation.

Note: each separate Matlab subroutine that is called within the Matlab code in the LabVIEW environment, must be included in the \Matlab\bin\win32 directory.

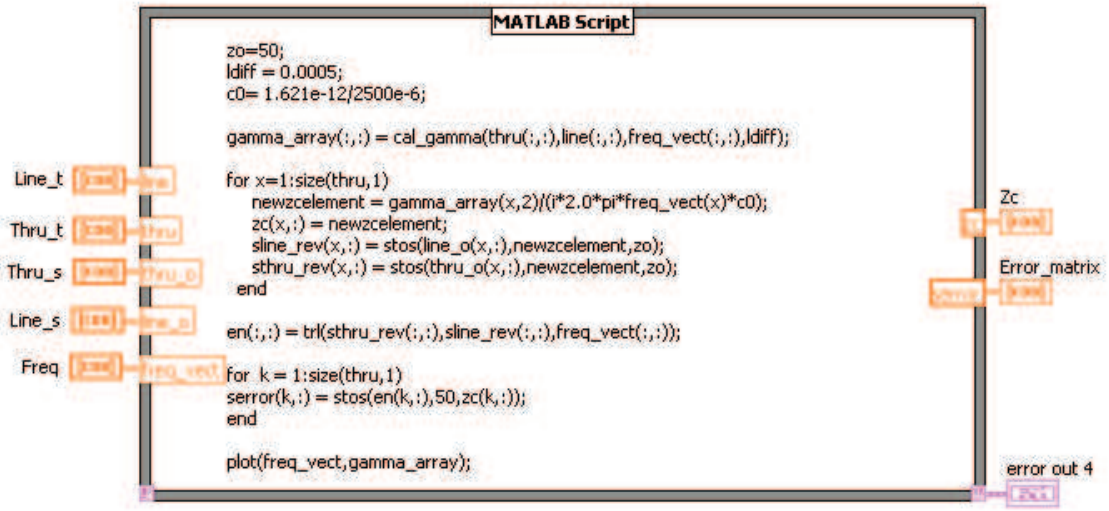


Figure 5.3: Block diagram for the second VI

5.3.2 Second Block

The VI in Figure. 5.3 is the second block which contains the Matlab script for calculation of the propagation constant γ, complex characteristic impedance Z_c and the error network parameters. The structure and principles for the VI block diagram implementation have been explained in the previous section. The naming convention for the controls and indicators is the same through out the different VIs, thus, indicating which outputs of the previous VI serve as inputs for the next VI.

The S-parameters for the through and line structures and T-parameters for the same, serve as controls or inputs, and the complex characteristic impedance Z_c and the error network parameters are the indicators or outputs for this VI. An icon-

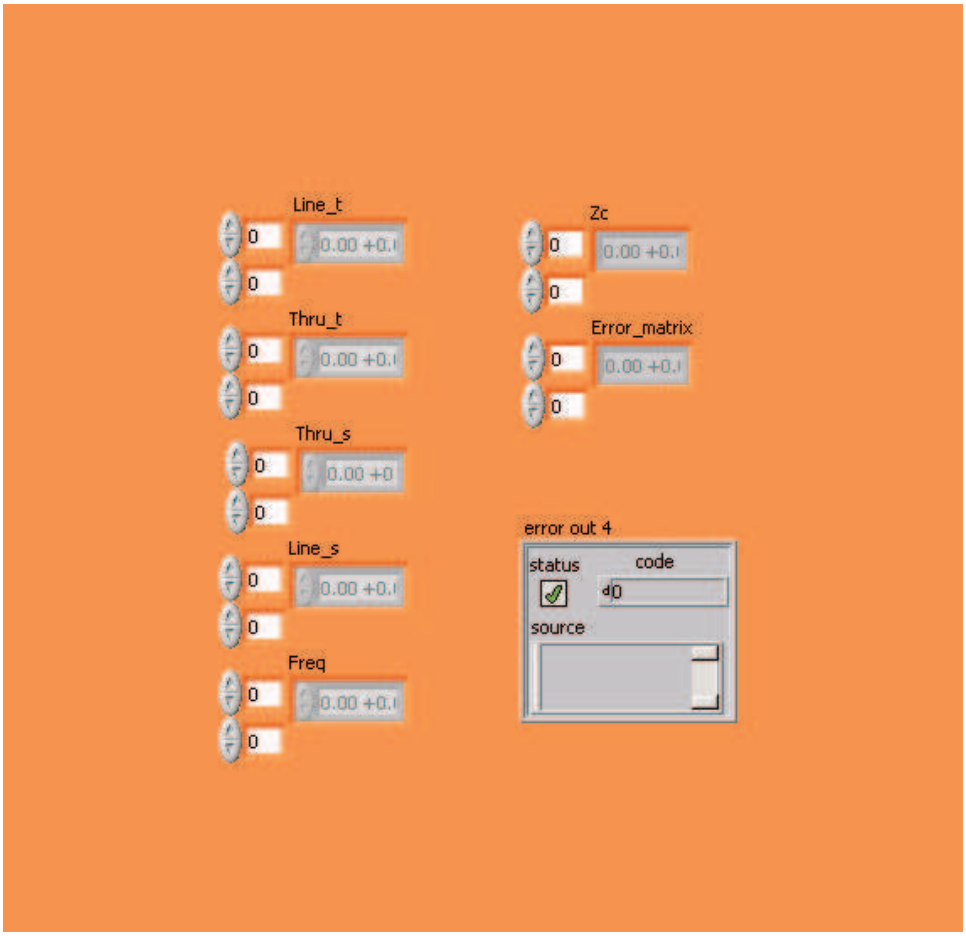


Figure 5.4: Front Panel for VI in Figure 5.3.

connector pane has been created for the VI so that it can be used as a subVI. The front panel for this VI is given by Figure 5.4.

5.3.3 Third Block

The Third block is similar to the first VI. It has the same structure and reads in the measured S-parameters for the DUT. The input data in the text file is in the magnitude-angle format. This data is converted into real-imaginary format for all measurement frequencies and then fed to the output of the VI's block diagram. An icon-connector pane has been created for the VI so that it can be used as a subVI.

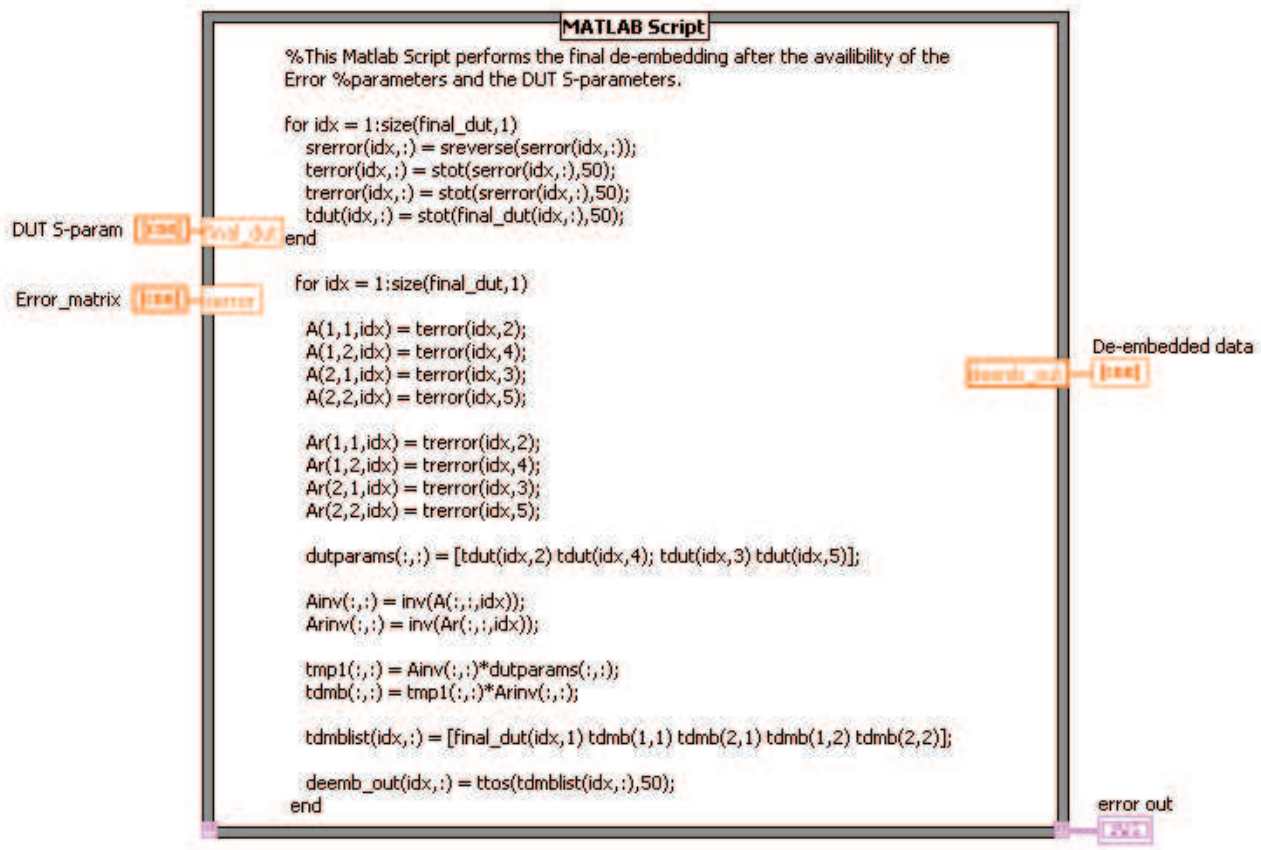


Figure 5.5: Block diagram for the fourth VI

5.3.4 Fourth Block

The VI in Figure. 5.5 is the fourth and final VI of the de-embedding implementation. It has the Matlab script that performs the final de-embedding after the availability of the error network parameters and the measured DUT S-parameters, which are the controls of the VI on the front panel. The output of this VI is the de-embedded S-parameters for the two-port network DUT for all measurement frequencies, which is an indicator on the front panel. An icon-connector pane has been created for the VI so that it can be used as a subVI.

The front panel for this VI is given by Figure 5.6.

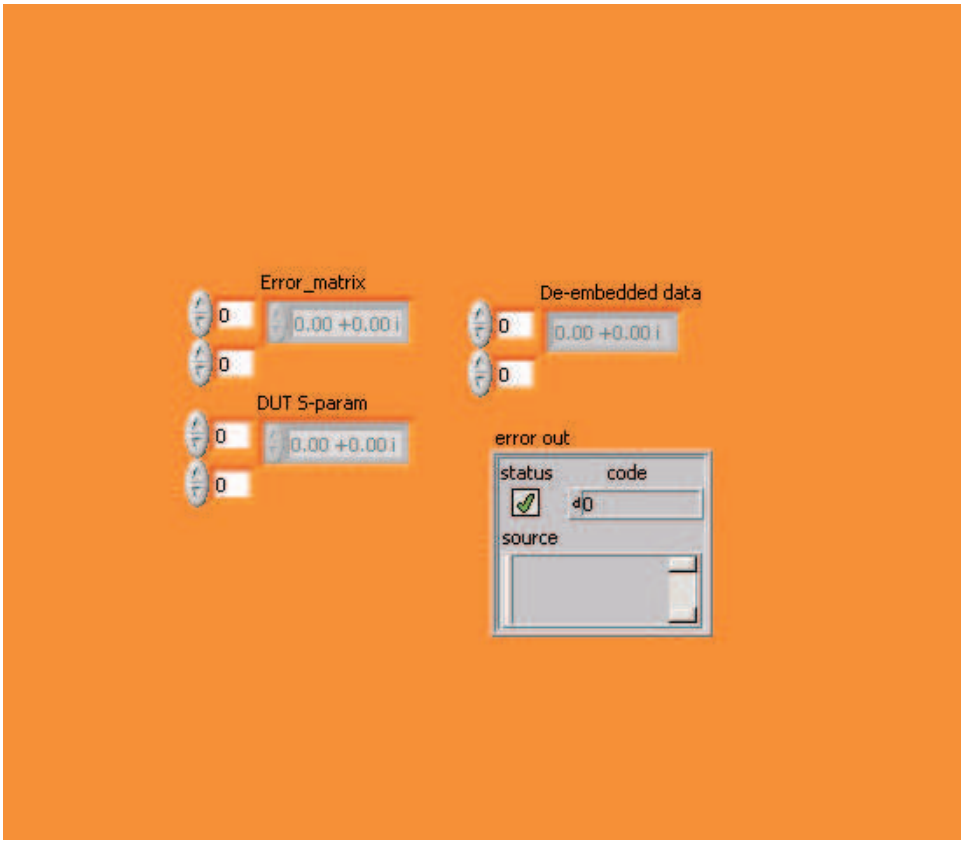


Figure 5.6: Front Panel for VI in Figure 5.5.

5.4 Summary

In this chapter, the implementation issues and details of the four main blocks for TL calibration has been provided. The above explained four VIs are integrated in the same order to implement a larger VI that essentially performs the TL De-embedding. Moreover, all the above VIs are used as subVIs in the final VI. The final VI will have its own front-panel. The reader is directed to Appendix B for information about the LabVIEW user's manual along with a screenshot of the front panel of the final VI.

Chapter 6

Results

6.1 Introduction

In this chapter we discuss the results obtained using the Through-Line algorithm for calibration and de-embedding. This report documents the transmission line and capacitance measurements made on a test wafer supplied by SEMATECH. Transmission lines were characterized using a Hewlett Packard network analyzer [7]. Capacitances were determined using conventional capacitance meter techniques [7]. Details of these measurements are explained in the following sections and have been previously reported [7] . The results shown here have been obtained by using NetA tools for de-embedding these microwave measurements for two port networks. The measurement set of one of the test wafer, has been taken.

6.2 Layers

The IC had three metallization layers as shown in Figure 6.1.

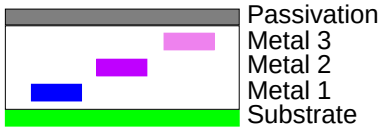


Figure 6.1: Cross-section of test IC showing three metallization layers.

6.3 Measurements

The measurements performed are capacitance and resistance measurements and microwave transmission line characterization. In general three forms of a structure were used to determine the characteristics of a measurement. Raw measurements were made on a long line (with the suffix l), a medium length line (with suffix m), and a short line (with suffix s). All derived parameters are reported for two extractions designated by the source of the raw data. Extractions with the suffix ls (or l,s) were extracted using the long and short lines. Extractions with the suffix lm (or l,m) were extracted using the long and medium lines.

6.4 Capacitance Measurement

Capacitance Measurements were made using the conventional contacting probes shown in Figure 3. There are two probes shown here; each has three contacts—a signal contact and two guard contacts. We use coaxial probes (GGB Model 40 picoprobes) which continue the coaxial probe to within 1 mm of the final test fingers. The guard contacts are extensions of the outer conductor of the coaxial line. On the chip, a set of three probe pads are contacted. The minimum dimension of the probe pads are 50 μ on one side with the outer pads typically connected to the chip ground. In previously reported measurements, GSG probes were used. Due to considerable probing difficulties encountered with the GSG probes, however, GS probes were used for the measurements reported here. Except for there being only one ground contact, the construction of the GS probes is identical to the construction of the GSG probes. Sub-picofarad capacitance measurements require a balanced probe system, but the electrical balance must, of course, be disturbed at the probe, resulting in a residual capacitance. The residual capacitance of the microprobe and the probe pad is approximately 70 fF; this must be subtracted out of the measurements. This establishes a resolution of 4 fF. Thus the resolution of the capacitance measurement is independent of that of the capacitance meter provided that the meter's resolution is 1 fF or less. Problems were encountered with the measurements. The principal problem was

planarity of the contact pads with the result that probe contact could not be made with the probes operating in their normal range of operation. The original design called for the ground and signal pads to be at the same level. However this was not achieved. Each set of measurements took two to three hours instead of the anticipated 15 minutes. Measurements were made on structures where possible but it is difficult to determine the accuracy of the measurements. However, the measurements were repeatable.

6.5 Transmission Line Characterization

A standard transmission line characterization was performed using a microwave network analyzer and microprobes, as in Figure 6.2. Again severe problems were encountered with the measurements. Measurements were made on structures where possible but it is difficult to determine the accuracy of the measurements. However the measurements were repeatable.

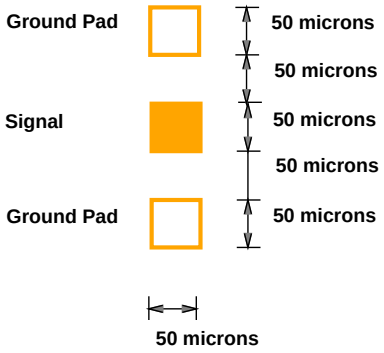


Figure 6.2: Single line microstrip fixture.

As a result of the planarity problems it was not possible to use Cascade probes because they are ceramic and cannot flex. The microprober used is shown in Figure 6.3, and a Hewlett Packard Automatic Network Analyzer (ANA) HP8510B connected to a HP workstation was used for data retrieval. The microstrip fixture used is shown

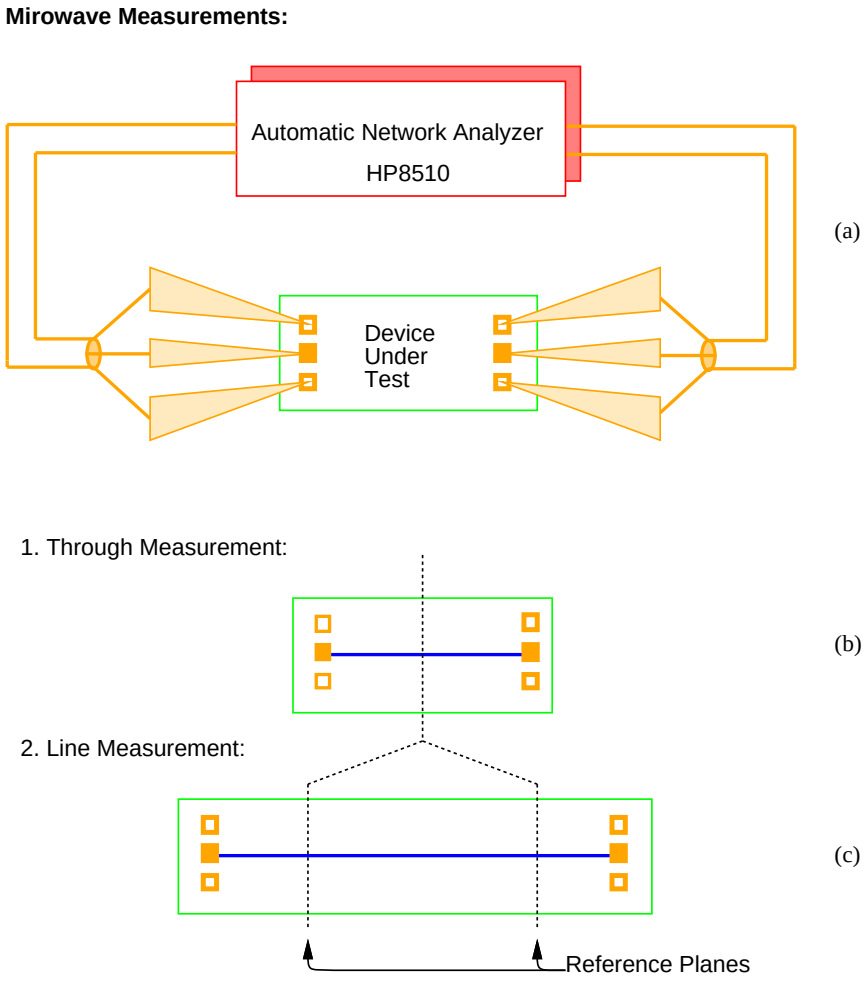


Figure 6.3: Microwave Measurement Set-up.

in Figure 6.2. It was not possible to make coupled line measurements because these require that contact be made with five conductors and severe problems were found with just three.

6.6 Results using NetA

The following results have been obtained using NetA tools for de-embedding two port microwave measurements. The medium line and the long line have been used as the through and the line standards for calibration. The results shown here are obtained

while using the long line as the DUT, de-embedded S-parameters of which are obtained using NetA. The details of the structures used as calibration standards are as mentioned below. These results have been matched with a previous characterization of the same structure in [7].

6.6.1 Single Line with Ground Plane

This structure is also known as the 2OG(S,M,L) structure or Single Line with Ground plane and three difference lengths of line. This is a unique name that has been provided at the time of measurements [7].

- Index: 2OG(S, M, L)
- Description: M2 line over M1 ground plane.
- Dimensions: $L_s = 400$ μm, $L_m = 800$ μm, $L_l = 6400$ μm; $W = W_m = 0.5$ μm

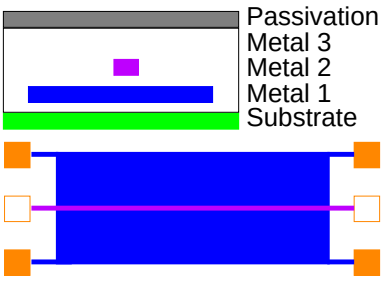


Figure 6.4: Cross-sectional and Plan view of the 2OG structure.

The following graphs show results for the medium and long line used as the through and line calibration standards. We have de-embedded the S-parameters for the long line.

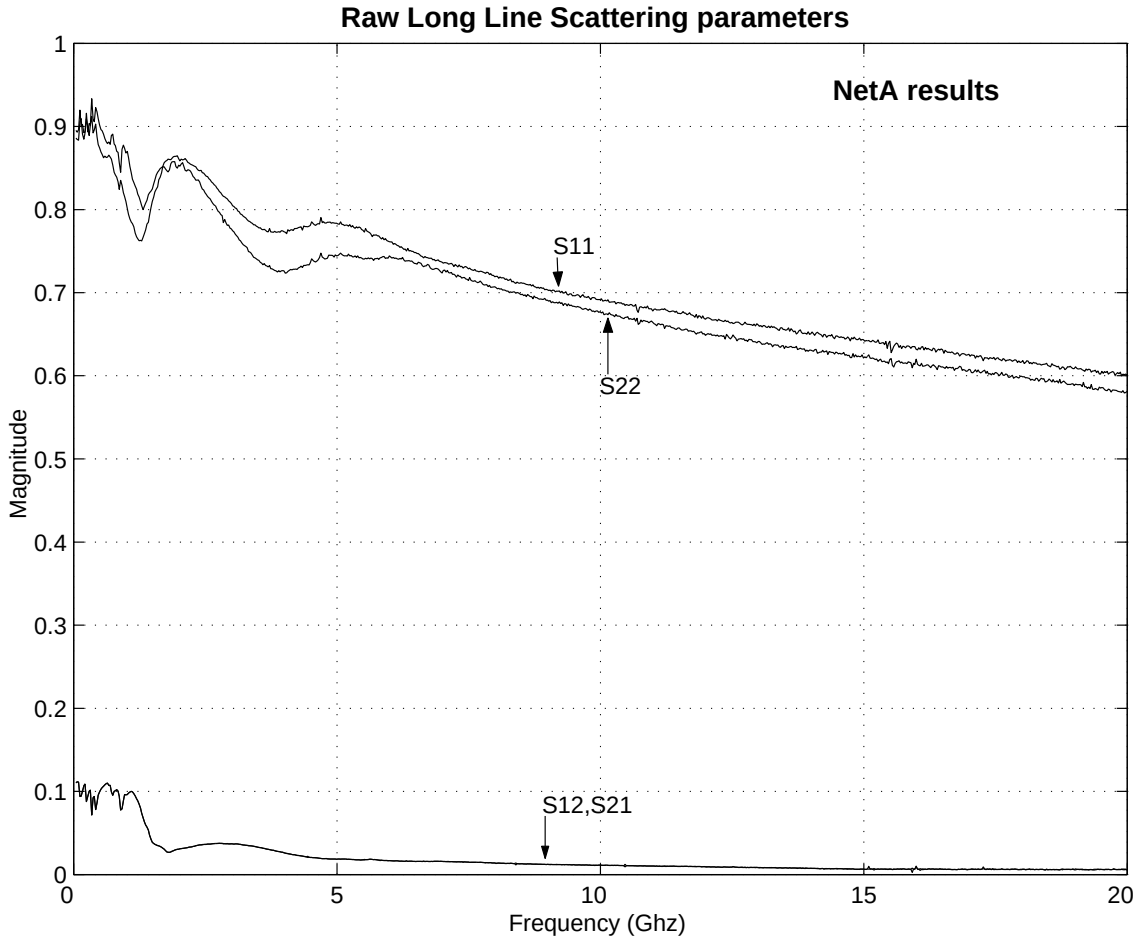


Figure 6.5: Raw S-parameters (magnitude) for the Long line.

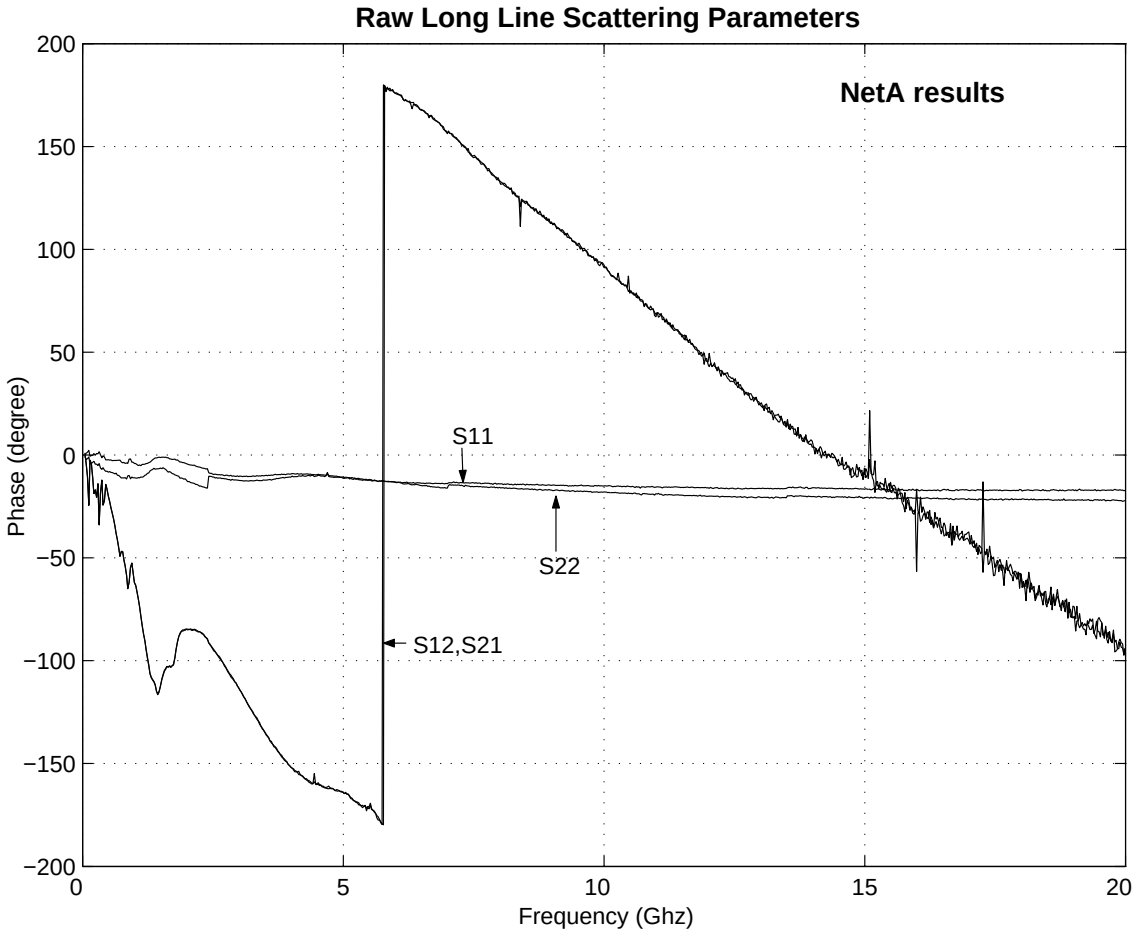


Figure 6.6: Raw S-parameters (phase) for the Long line.

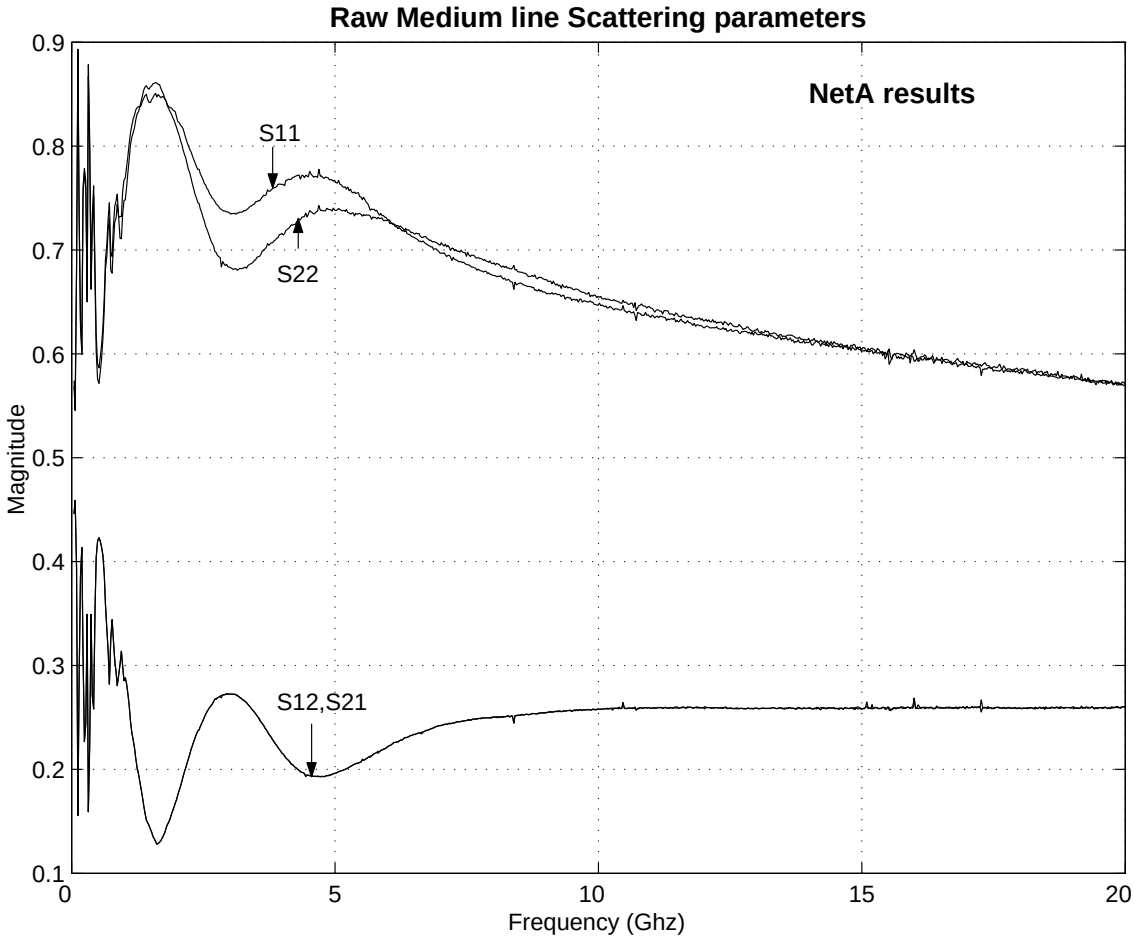


Figure 6.7: Raw S-parameters (magnitude) for the Medium line.

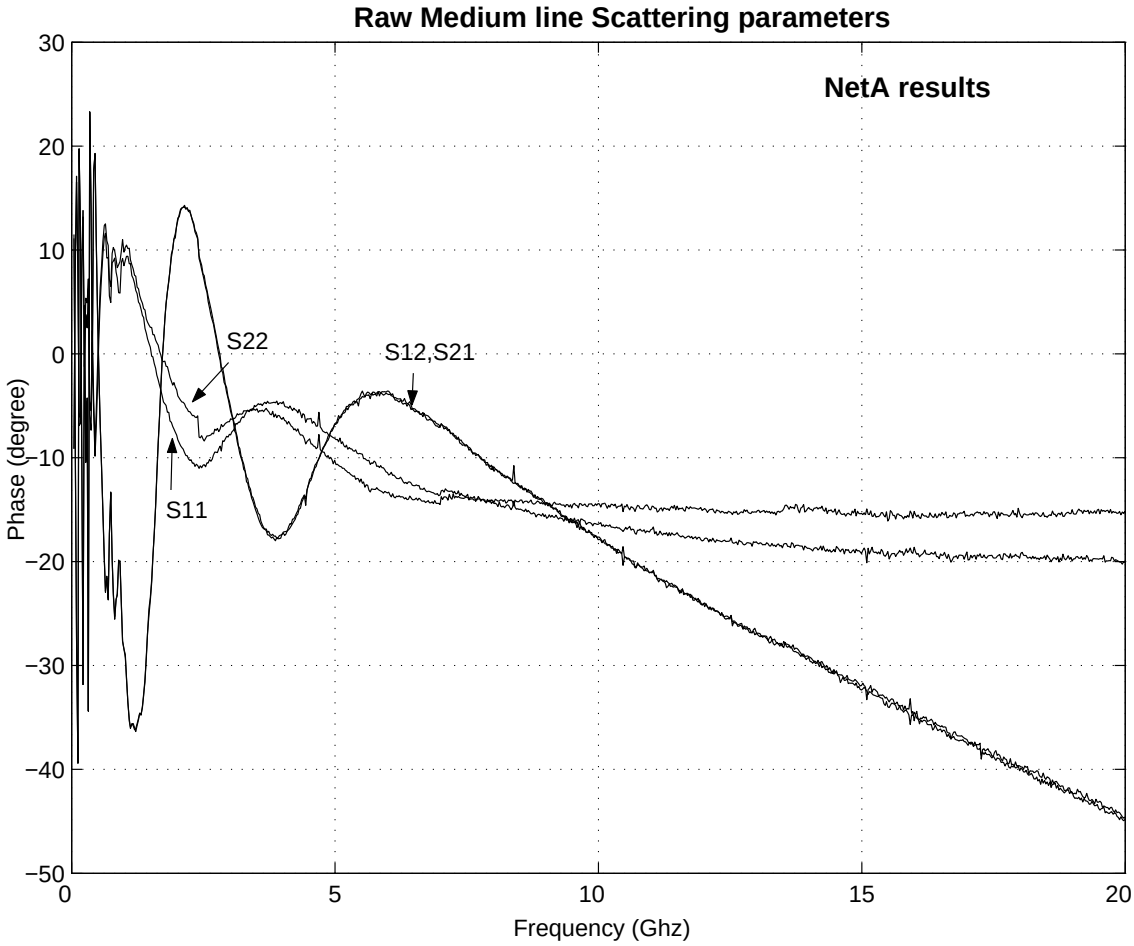


Figure 6.8: Raw S-parameters (phase) for the Medium line.

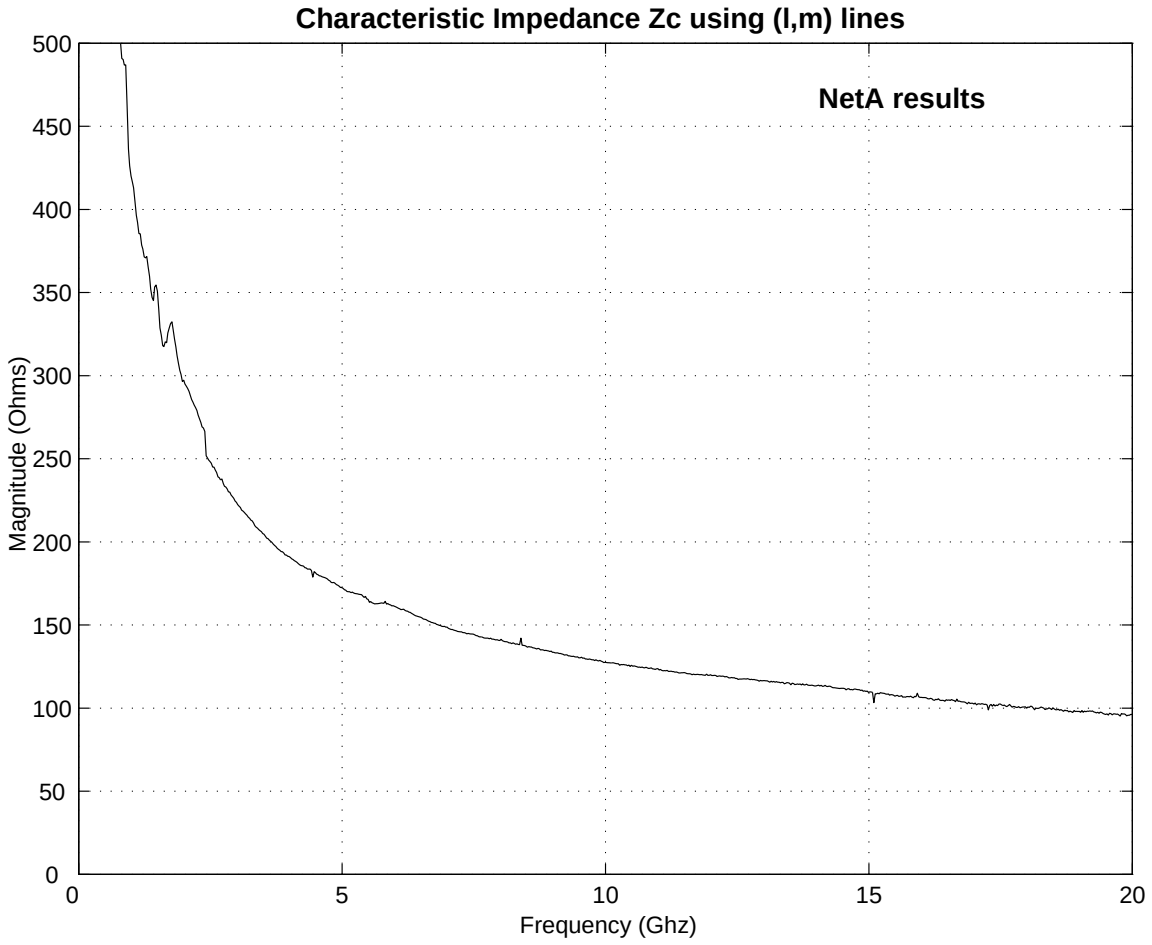


Figure 6.9: Characteristic impedance (magnitude) calculated using the ETRL algorithm.

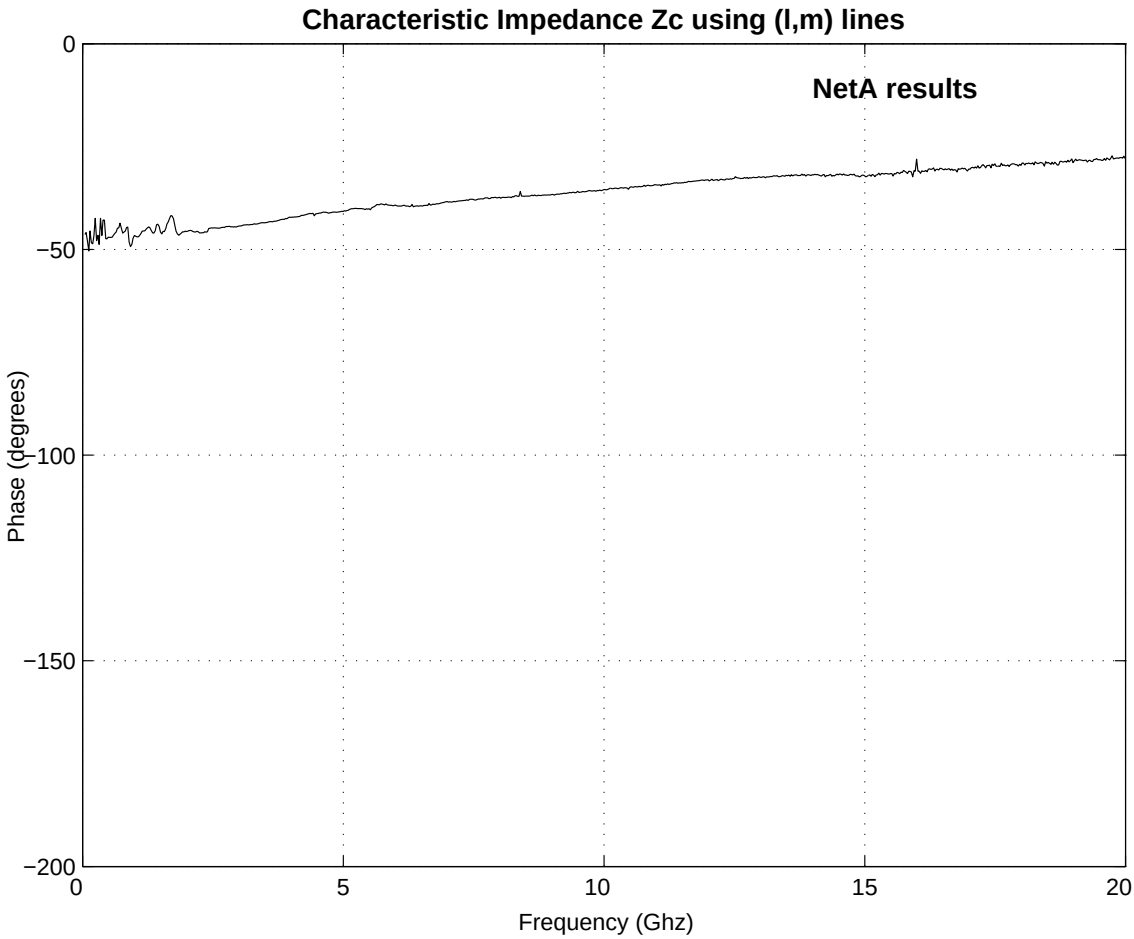


Figure 6.10: Characteristic impedance (phase) calculated using the ETRL algorithm.

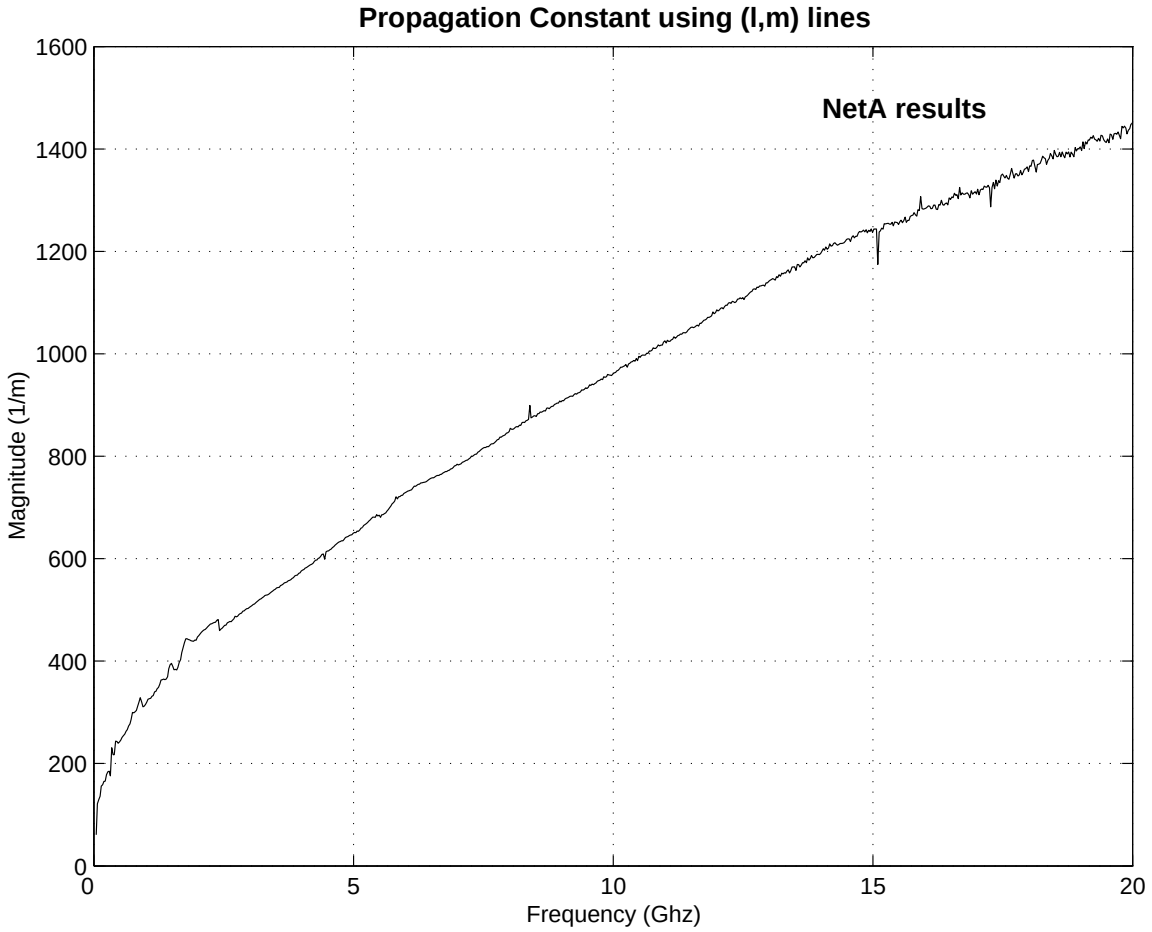


Figure 6.11: Propagation Constant (magnitude) using (l,m) lines.

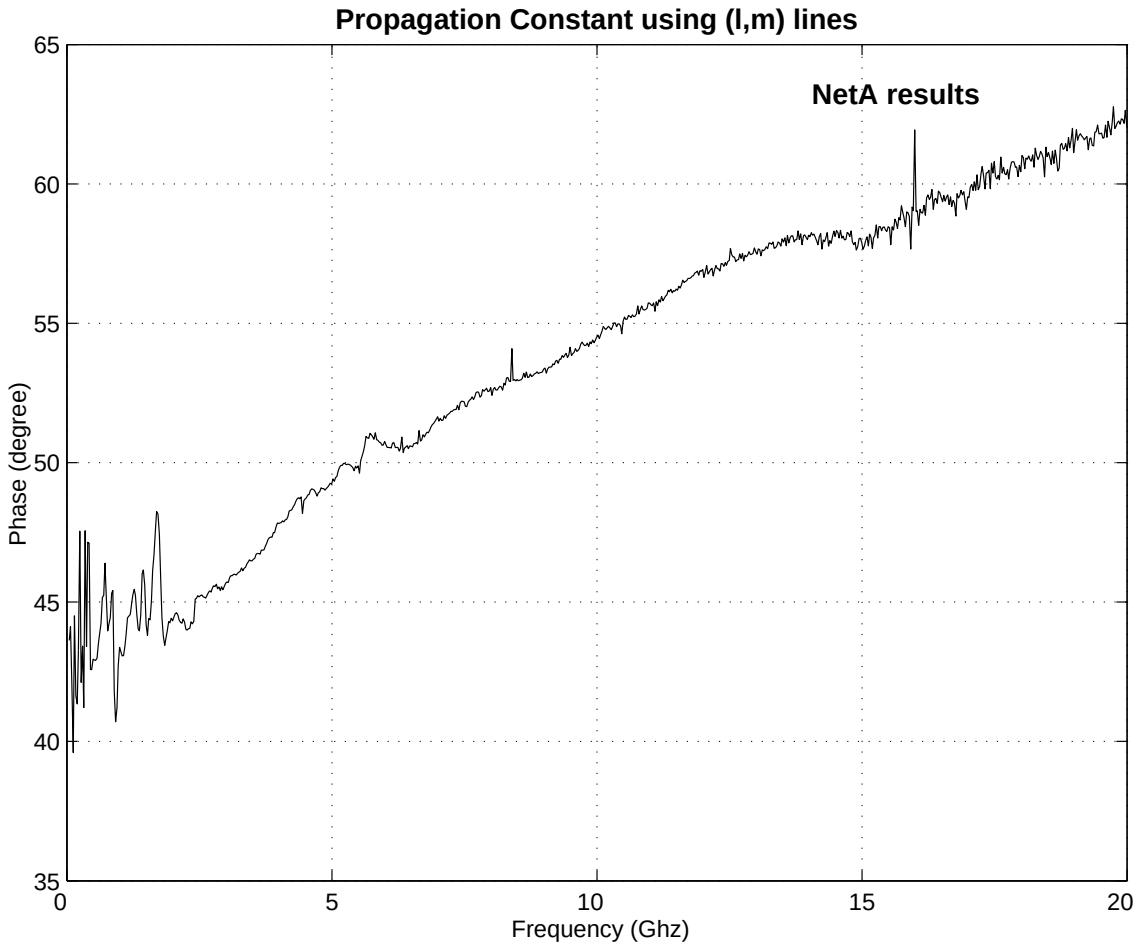


Figure 6.12: Propagation Constant (phase) using (l,m) lines.

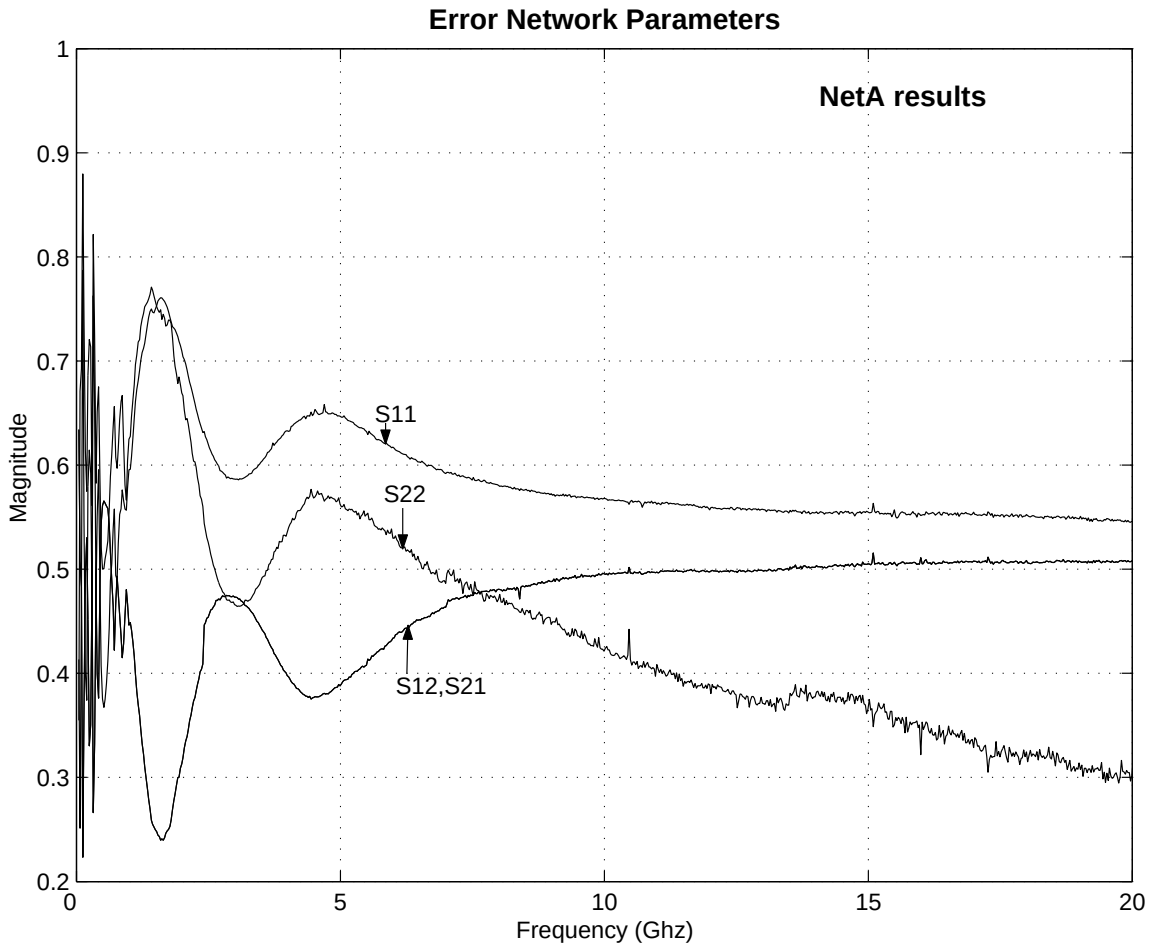


Figure 6.13: Error network S-parameters (magnitude) using (l,m) lines.

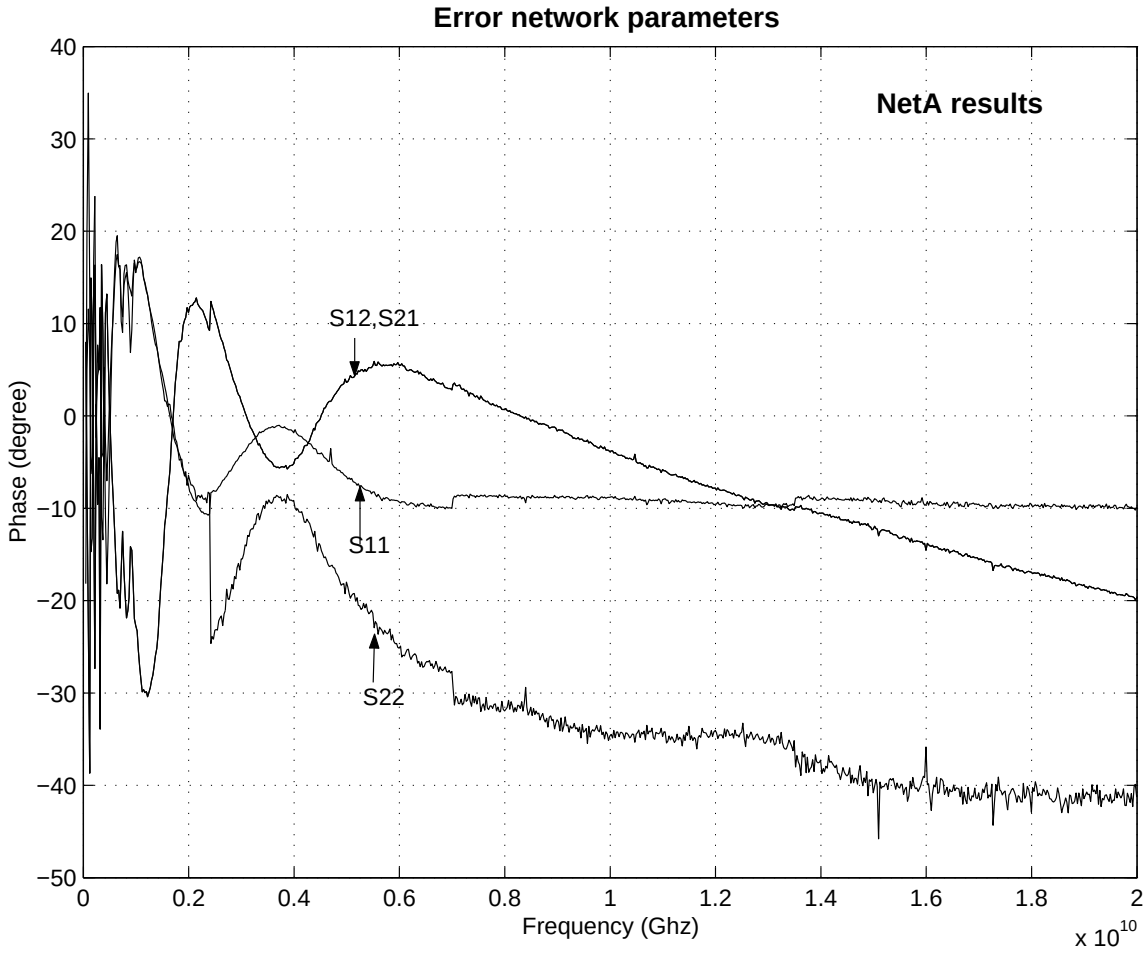


Figure 6.14: Error network S-parameters (phase) using (l,m) lines.

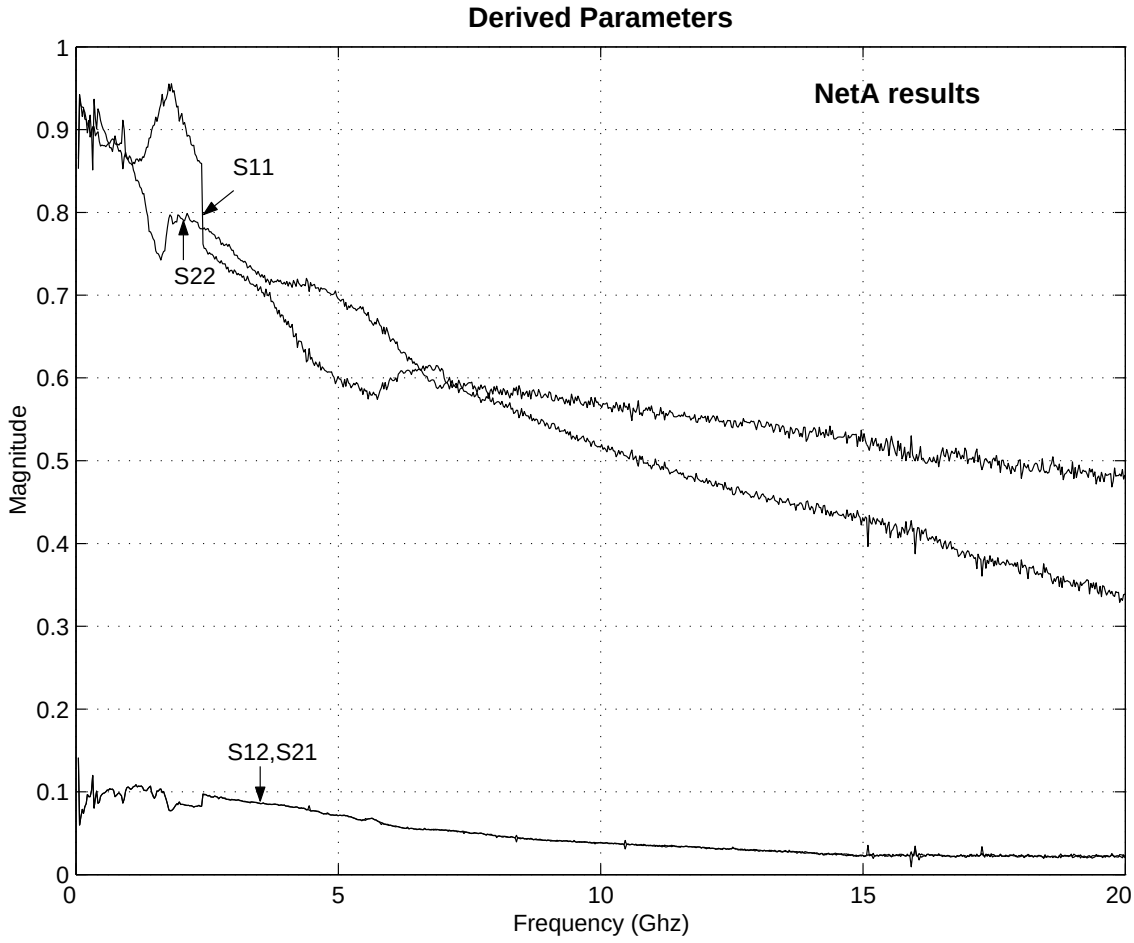


Figure 6.15: De-embedded S-parameters (magnitude) of the DUT.

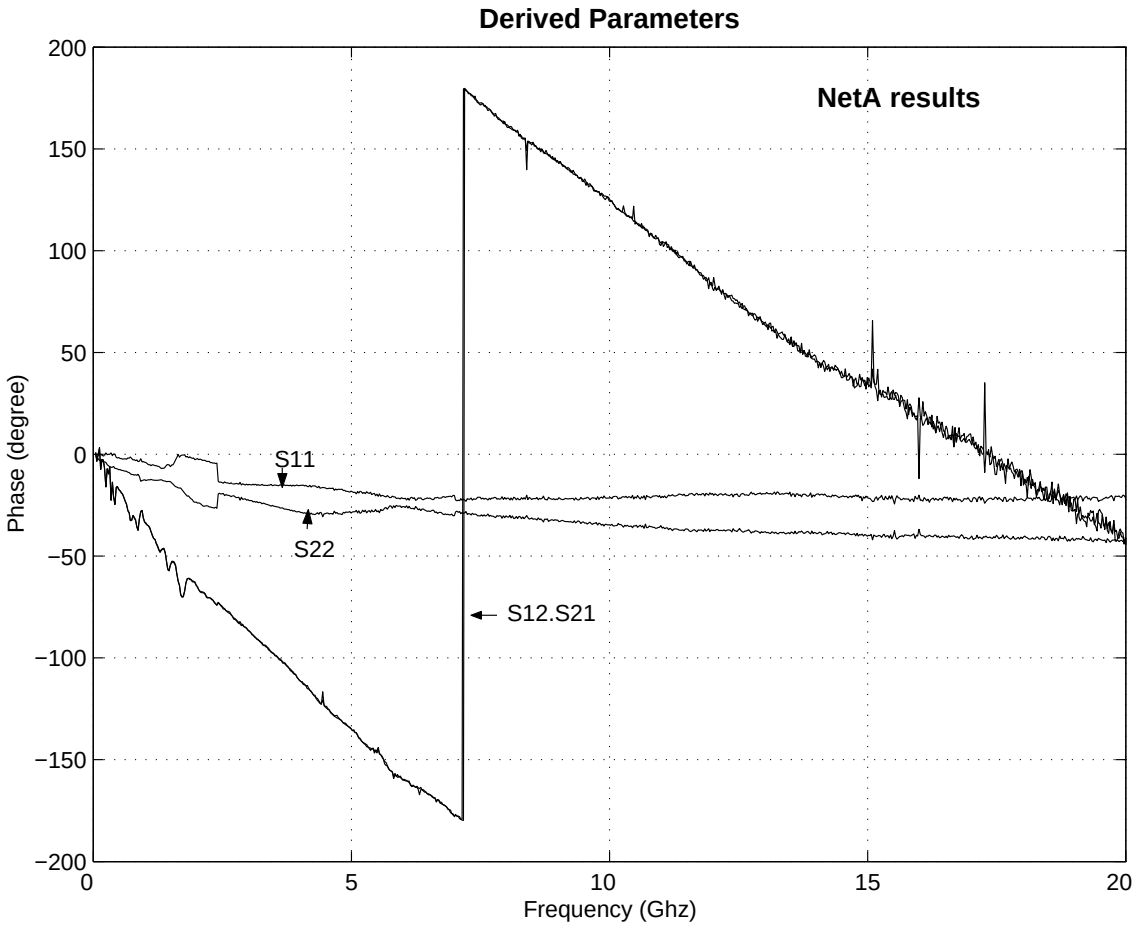


Figure 6.16: De-embedded S-parameters (phase) of the DUT.

Chapter 7

Conclusion

7.1 Summary

The objective of this study was to study symmetric methods of calibration and implement a Computer Aided tool (NetA) that can be used for the purposes of calibration and de-embedding as well as S-parameter processing. This work consists of two parts: Introduction of the symmetrical methods for calibration and then Novel implementations of these procedures.

In the first part of this study, we discussed symmetrical calibration methods that have been developed before. The symmetry condition helps us reduce the number of required standards for microwave calibration without physically inserting them. Using properties of first order symmetry and by synthesizing the reflection standard, the TL technique can be effectively used. TL was also enhanced by calculating the complex characteristic impedance of the line standard. For this, the ETRL method was used.

Using the above two calibration techniques, NetA tools for calibration of microwave measurements was developed. NetA's process flow was also explained. An extension to the implementation of NetA has been shown by implementing the calibration in LabVIEW. This implementation provides us with a capability of interfacing the calibration algorithm with the VNA and using it as a virtual instrument to calibrate and analyze data real–time. A previous measurement set has been taken and

simulated using NetA tools. These results have been shown.

7.2 Future Work

Additional areas of study include:

1. Analysis of the sensitivity of the TL calibration technique to randomness in transmission line parameters.
2. Symmetry sensitivity for different kind of transmission lines.
3. Further expansion of NetA to larger port devices.

Bibliography

[1] G. F. Engen and C. A. Hoer, "Thru-reflect-line: an improved technique for calibrating the dual six-port automatic network analyzer," *, IEEE Trans. Microwave Theory and Techniques,* pp. 987–993, Dec. 1979.

[2] M. B. Steer, S. B. Goldberg, G. Rinne, P. D. Franzon, I. Turlik and J. Kasten, "Introducing the through-line de-embedding procedure," *Int. Microwave Symposium,* pp. 1455–1458, June 1992.

[3] M. B. Steer, S. B. Goldberg, P. D. Franzon and J. Kasten, "Experimental electrical characterization of interconnects and discontinuities in high-speed digital systems," *IEEE Trans. Components, Hybrids and Manufacturing Technology,* pp. 761–765, Dec 1991.

[4] J. S. Kasten, M. B. Steer and R. Pomerleau, "Enhanced through-reflect-line characterization of two-port measuring systems using free-space capacitance calibration," *IEEE Trans. Microwave Theory and Techniques,* pp. 215-217, Feb 1990.

[5] J.P. Mondal, T. Chen, "Propagation Constant Determination in Microwve Fixture De-embedding Procedure," *EEE Trans. Microwave Theory and Techniques,* pp. 706-714, Apr. 1988.

[6] J.S. Kasten, *Calibration of Automatic Network Analysis and Computer Aided Microwave Measurement,* Masters Thesis, North Carolina State University, 1992.

[7] M. B. Steer, P. D. Franzon, W. J. Ficken, A. W. Glaser, B. Biswas and S. Lipa, *Experimental Determination of On-Chip Interconnect Capacitances, 1998 SEMATECH Test Report, 2nd Edition,* pp. 4–7, pp. 121–126.

[8] *Applying Error Correction to Network Analyzer Measurements, Application Note, AN 1287-3,* pp. 1–15, Agilent Technologies.

[9] *Matlab User Guide.*

[10] *LabVIEW User Guide.*

Appendix A

NetA Programmer's Manual

This appendix is a user's manual for NetA. Firstly the user is informed about the command syntax to be used and the series in which the TL NetA routines are to be executed for calibration and de-embedding. Secondly, all the utility routines that can be used for the purpose of data analysis are explained. The routines that are to be executed at the MATLAB command prompt for the TL calibration are:

1. GET_MAG_ANG
2. TOUCHSTONE
3. TL_CALIB
4. DUT and FINAL
5. TSL

The usage and purpose of each of the above routines is explained in the coming sections.

A.1 get_mag_ang

Purpose : Gets the raw S-parameters of the through and the line calibration standards.
Output Format : Magnitude and Phase
Array Ordering : freq S11 S21 S12 S22
Usage : get_mag_ang;
Code :

```
thru_data = xlsread(['short.xls']);

line_data = xlsread(['long.xls']);

freq_vect = xlsread(['freq.xls']);
```

The raw data is now available in local variables. These variables will be used to pass data to the next routine and will also show up in the *Workspace* area of MATLAB as an array with its dimensions. The short.xls, long.xls and freq.xls are excel files. When read into NetA, the files must only contain numeric data. The file freq.xls contains just one column of all measurement frequencies. Usage of this vector is minimum throughout the code and can be removed completely if needed, as the frequency vector is the first column in all data arrays.

A.2 dut and final

Purpose: We use this routine to read in the S-parameters for the DUT to be de-embedded.

Output Format: Real–Imaginary.

Array Ordering: freq S11 S21 S12 S22.

Code:

```
dut = xlsread(['long.xls']);

final_dut = final(dut);

The FINAL sub-routine:

function retval = final(dut)

for k = 1:size(dut,1)
     rawdut(1,1) =ptor(dut(k,2),dut(k,3));
     rawdut(2,1) =ptor(dut(k,4),dut(k,5));
     rawdut(1,2) =ptor(dut(k,6),dut(k,7));
     rawdut(2,2) =ptor(dut(k,8),dut(k,9));

     final_dut(k,:) = [dut(k,1) rawdut(1,1) rawdut(2,1) rawdut(1,2) rawdut(2,2)];
 end


retval = final_dut(:,:);
```

We first read the raw DUT S-parameters and then convert them to the Real–Imaginary format.

A.3 tl_calib

Purpose: Calculates the propagation constant, characteristic impedance and the error matrix.

Output Format: Real–Imaginary.

Array Ordering: freq S11 S21 S12 S22 for the error network.

freq γ for the propagation constant.

Zc for the complex characteristic impedance.

Usage: [serror,gamma_array,zc] = tl_calib(thru,line,freq_vect,ldiff,c0,thru_o,line_o);

Code:

```
function [serror,gamma_array,zc] =
tl_calib(thru,line,freq_vect,ldiff,c0,thru_o,line_o) zc=[]; zo=50;

gamma_array = cal_gamma(thru,line,freq_vect,ldiff);

for x=1:size(thru,1)
     newzcelement = gamma_array(x,2)/(i*2.0*pi*freq_vect(x)*c0);
     zc =  cat(1,zc,newzcelement);
     sline_rev(x,:) = stos(line_o(x,:),newzcelement,zo);
     sthru_rev(x,:) = stos(thru_o(x,:),newzcelement,zo);
 end

en(:,:) = trl(sthru_rev(:,:),sline_rev(:,:));

for x=1:size(thru,1)
    serror(x,:) = stos(en(x,:),50,zc(x,:));
end

gamma_array(:,:); zc(:,:);
```

We now have the error matrices, propagation constant and characteristic impedance arrays built by using this routine. All of these arrays will be produced in the Workspace area of MATLAB. We have now, completed the TL Calibration and can use the data for de-embedding.

A.4 touchstone

Purpose: Converts the raw data from Polar to Rectangular co-ordinates, symmetrizes the data and calculates the T-parameters of the calibration structures.

Output Format: Real–Imaginary.

Array Ordering: freq S11 S21 S12 S22.

Usage: [thru,line,thru_o,line_o] = touchstone(thru_data,line_data,freq_vect);

Code:

```
function [thru,line,thru_o,line_o] =
touchstone(thru_data,line_data,freq_vect) z0 =50;

if (nargin<3)
    error(' Input Arguments are less....');
end

for k = 1:size(thru_data,1)
       v = thru_data(k,:);
       t11 = v(2);
       p11 = v(3);
       t21 = v(4);
       p21 = v(5);
       t12 = v(6);
       p12 = v(7);
       t22 = v(8);
       p22 = v(9);

       w = line_data(k,:);
       L11 = w(2);
       P11 = w(3);
       L21 = w(4);
       P21 = w(5);
       L12 = w(6);
       P12 = w(7);
       L22 = w(8);
       P22 = w(9);


    sthru11 =(ptor(t11,p11)+ptor(t22,p22))/2 ;
    sthru12 =(ptor(t21,p21)+ptor(t12,p12))/2;
```

```
  sthru21 =sthru12;
  sthru22 =sthru11;


  sline11 =(ptor(L11,P11)+ptor(L22,P22))/2;
  sline12 =(ptor(L21,P21)+ptor(L12,P12))/2;
  sline21 =sline12;
  sline22 =sline11;

  thru_o(k,:) = [freq_vect(k,1) sthru11 sthru21 sthru12 sthru22];
  line_o(k,:) = [freq_vect(k,1) sline11 sline21 sline12 sline22];

  thru(k,:) = stot(thru_o(k,:),z0);
  line(k,:) = stot(line_o(k,:),z0);
end
```

This routine produces the S-parameters and the T-parameters of the calibration standards and stores them in local variables. At the command prompt, also enter values for the difference in the line and through standard lengths as *ldiff* and the free-space-capacitance $c0$. The arrays produced by this routine, *ldiff* and C_o are used as input arguments for the TL calibration routine.

A.5 tsl

Purpose: Performs the TL de-embedding once the DUT S- parameters and the error network parameters are available.

Output Format: Real–Imaginary.

Array ordering: freq S11 S21 S12 S22.

Usage: deemb_dut = tsl(serror,final_dut);

Code:

```
function retval = tsl(serror,final_dut)

if (nargin<2)
    error ('Input arguments are less..');
end

for idx = 1:size(final_dut,1)
    srerror(idx,:) = sreverse(serror(idx,:));
    terror(idx,:) = stot(serror(idx,:),50);
    trerror(idx,:) = stot(srerror(idx,:),50);
    tdut(idx,:) = stot(final_dut(idx,:),50);
end

for idx = 1:size(final_dut,1)

    A(1,1,idx) = terror(idx,2);
    A(1,2,idx) = terror(idx,4);
    A(2,1,idx) = terror(idx,3);
    A(2,2,idx) = terror(idx,5);

    Ar(1,1,idx) = trerror(idx,2);
    Ar(1,2,idx) = trerror(idx,4);
    Ar(2,1,idx) = trerror(idx,3);
    Ar(2,2,idx) = trerror(idx,5);

    dutparams(:,:) = [tdut(idx,2) tdut(idx,4); tdut(idx,3) tdut(idx,5)];

    Ainv(:,:) = inv(A(:,:,idx));
    Arinv(:,:) = inv(Ar(:,:,idx));

    tmp1(:,:) = Ainv(:,:)*dutparams(:,:);
```

```
    tdmb(:,:) = tmp1(:,:)*Arinv(:,:);

    tdmblist(idx,:) = [final_dut(idx,1) tdmb(1,1) tdmb(2,1) tdmb(1,2) tdmb(2,2)];

 end
end

 retval = ttos(tdmblist,50);
```

Now, the TL de-embedding is complete and the result is available as the variable *deemb_dut* in the work space area of MATLAB. The later sections will explain the utility routines that have been internally called within the above mentioned routines and which are also available for later usage.

A.6 cal_gamma

Purpose: Calculates the complex propagation constant γ.

Output Format: Real–Imaginary.

Array Ordering: freq γ

Usage: This routine is called internally from within the *tl_calib* routine.

Code:

```
function retval = cal_gamma(thru,line,freq,L)

if (nargin<4)
    error('Input arguments are less...');
end

piby2 = pi/2;
Z0=50;

c0= 1.621e-12/2500e-6;
  twobetac = 2.0*pi/L;
  adflag = 0;
  adjust = 0;
  betam1 = 100.0;
  betam2 = 100.0;
  betam3 = 100.0;
  betam4 = 100.0;
  betam5 = 100.0;
  g_ans=[];

for k = 1:size(thru,1) A(k,1)= (thru(k,2)*line(k,5) +
line(k,2)*thru(k,5)) - (thru(k,4)*line(k,3) +
thru(k,3)*line(k,4));

phase(k,1) = angle(A(k,1)); A2m4(k,1) = (A(k,1)^2 - 4);

ap(k,1) = (A(k,1) + sqrt(A2m4(k,1)))/2; an(k,1) = (A(k,1) -
sqrt(A2m4(k,1)))/2;

if     ( phase(k,1) > 0 & phase(k,1) < piby2)
    egamma(k,1) = ap(k,1);
elseif ( phase(k,1) > piby2 & phase(k,1) < pi)
    egamma(k,1) = an(k,1);
```

```
elseif ( phase(k,1) > -pi & phase(k,1) < -piby2)
    egamma(k,1) = an(k,1);
elseif ( phase(k,1) > -piby2 & phase(k,1) < 0)
    egamma(k,1) = ap(k,1);
end

 int_gam(k,1) = log(egamma(k,1));
 final_gamma(k,1) = int_gam(k,1)/L;


 beta(k,1) = imag(final_gamma(k,1));

 if ((beta(k,1) < 0) & (adflag == 0) & (abs(beta(k,1)) > twobetac*9.0/20.0))
  adjust = twobetac;
  adflag = 1;

elseif ((betam5 < 0) & (betam4 < 0) & (betam3 < 0) & (betam2 < 0)
& (betam1 < 0) & (beta > 0))
         adflag = 0;
       end
        betam5 = betam4;
        betam4 = betam3;
        betam3 = betam2;
        betam2 = betam1;
        betam1 = beta(k,1);
        g_ans(k,1) = real(final_gamma(k,1))+i*abs(beta(k,1)+adjust);

    end


gamma_int = cat(2,freq,g_ans);
retval = gamma_int;
```

A.7 tl

Purpose: Calculates the error network parameters for the two port TL calibration.

Output Format: Real–Imaginary.

Array ordering: freq S11 S21 S12 S22

Usage: This routine is called internally from within the *tl_calib* routine.

Code:

```
function retval = trl(thru,line)

rthru = stor(thru); rline = stor(line);

freq_vect =xlsread(['freq.xls']);
rerror = [];

      for idx=1:size(thru,1)

         gammasc = thru(idx,2) - thru(idx,3);
         rt = [rthru(idx,2) rthru(idx,4); rthru(idx,3) rthru(idx,5)];
         rl = [rline(idx,2) rline(idx,4); rline(idx,3) rline(idx,5)];

         t = rl * inv(rt);
         tmpa = t(2,1);
         tmpb = t(2,2)-t(1,1);
         tmpc = -t(1,2);

         a1=(-tmpb+sqrt(tmpb^2-4*tmpa*tmpc))/(2*tmpa);
         a2=(-tmpb-sqrt(tmpb^2-4*tmpa*tmpc))/(2*tmpa);

         if (abs(a2) > abs(a1))
            ac = a2;
            b = a1;
         else
            ac = a1;
            b = a2;
         end

         w1 = gammasc;
         a = (w1-b)/(-1.0*(1.0-w1/ac));
         c = a/ac;
```

```
      r22 =sqrt(-thru(idx,2)/(thru(idx,3)*(a*c-b)));

      rerror(idx,:) = [freq_vect(idx,1) a*r22 c*r22 b*r22 r22];
      final_rerror(idx,:) = rtos(rerror(idx,:));
  end

retval = final_rerror(:,:);
```

A.8 ptor

Purpose: Converts polar to rectangular.

Output Format: Real–Imaginary.

Usage: out_var = ptor(mag,ang);

Code:

```
function retval = ptor (x,y)
r3 = (x*cos(y));
r4 = (x*sin(y));
final  = r3+r4*i;
end

retval = final;
```

A.9 stot

Purpose: Converts the S-parameters to ABCD or T-parameters.

Output Format: Real–Imaginary.

Array Ordering: freq S11 S21 S12 S22

Usage: out_var = stot(input_arg,z0);

Code:

```
function retval = stot (A,z0)
  retval = 0;
  if (nargin < 1)
    error ('stot (two-port s param list,[Zo])');
end


    for idx = 1:size(A,1)
       v = A(idx,:);
       s11 = v(2);
       s21 = v(3);
       s12 = v(4);
       s22 = v(5);

       d = s21+s21;
```

```
    t11 = ((1+s11)*(1-s22)+s12*s21)/d;
    t12 = z0*((1+s11)*(1+s22)-s12*s21)/d;
    t21 = ((1-s11)*(1-s22)-s12*s21)/(z0*d);
    t22 = ((1-s11)*(1+s22)+s12*s21)/d;

    t(idx,:) = [A(idx,1) t11 t21 t12 t22];
end

retval = t;
```

A.10 stor

Purpose: Converts the S-parameters to the R-parameters or wave cascading parameters.

Output Format: Real–Imaginary.

Array Ordering: freq r11 r21 r12 r22

Usage: out_var = stor(input_arg);

Code:

```
function retval = stor(A)

 r = [];
 for idx = 1:size(A,1)
    s11 = A(idx,2);
    s21 = A(idx,3);
    s12 = A(idx,4);
    s22 = A(idx,5);

    r22 = 1/s21;
    r21 = -s22/s21;
    r12 = s11/s21;
    r11 = s12-s11*s22/s21;

    r(idx,:) = [A(idx,1) r11 r21 r12 r22];
end

 retval = r;
```

A.11 stoy

Purpose: Converts the S-parameters to the Y-parameters.

Output Format: Real–Imaginary.

Array Ordering: freq y11 y21 y12 y22

Usage: out_var = stoy(A,z0);

Code:

```
function retval = stoy (A,z0)
  retval = 0;
  if (nargin < 1)
    error ('stoy (s parameter list,[Zo])');
end

  y0 = 1/z0;

    v = A(1,:);
    s11 = v(2);
    s21 = v(3);
    s12 = v(4);
    s22 = v(5);

    for idx = 1:size(A,1)
       v = A(idx,:);
       s11 = v(2);
       s21 = v(3);
       s12 = v(4);
       s22 = v(5);

       d = (1+s11)*(1+s22)-s12*s21;

       y11 = y0*((1-s11)*(1+s22)+s12*s21)/d;
       y12 = y0*(-2*s12)/d;
       y21 = y0*(-2*s21)/d;
       y22 = y0*((1+s11)*(1-s22)+s12*s21)/d;

       y(idx,:) = [A(idx,1) y11 y21 y12 y22];

   end
   retval = y;
```

A.12 stoz

Purpose: Converts the S-parameters to the Z-parameters.

Output Format: Real–Imaginary.

Array Ordering: freq z11 z21 z12 z22

Usage: out_var = stoz(A,z0);

Code:

```
function retval = stoz (A,z0)
  retval = 0;
  if (nargin < 1)
    error ('stoz (two-port s parameter list,z0)');
end

z = [];
    for idx = 1:size(A,1)
       v = A(idx,:);
       s11 = v(2);
       s21 = v(3);
       s12 = v(4);
       s22 = v(5);

       d = (1-s11)*(1-s22)-s12*s21;

       z11 = z0*((1+s11)*(1-s22)+s12*s21)/d;
       z12 = z0*(s12+s12)/d;
       z21 = z0*(s21+s21)/d;
       z22 = z0*((1-s11)*(1+s22)+s12*s21)/d;

       z(idx,:) = [A(idx,1) z11 z21 z12 z22];

   end

    retval = z;
```

A.13 sreverse

Purpose: Reverses the input matrix. Output Format: Real–Imaginary.

Array Ordering: freq s22 s21 s12 s11

Usage: out_var = sreverse(A);

Code:

```
function retval = sreverse (A)
 rev = [];
    for idx = 1:size(A,1)
       rev = [A(idx,1) A(idx,5) A(idx,3) A(idx,4) A(idx,2)];
    end
 retval = rev;
```

A.14 stoh

Purpose: Converts the S-parameters to the H-parameters.

Output Format: Real–Imaginary.

Array Ordering: freq h11 h21 h12 h22

Usage: out_var = stoh(A,z0);

Code:

```
function retval = stoh (A,z0)
if (nargin < 1)
    error ('stoh (two-port s parameter list,z0)');
end

if (nargin < 2)
    z0 = 50;
  end
z = [];

    for idx = 1:size(A,1)
       v = A(idx,:);
       s11 = v(2);
       s21 = v(3);
       s12 = v(4);
       s22 = v(5);
```

```
        d = (1-s11)*(1+s22)+s12*s21;

        h11 = z0*((1+s11)*(1+s22)-s12*s21)/d;
        h12 = z0*(s12+s12)/d;
        h21 = -z0*(s21+s21)/d;
        h22 = z0*((1-s22)*(1-s11)-s12*s21)/d;

        z(idx,:) = [A(idx,1) h11 h21 h12 h22];
    end

    retval = z;
```

A.15 magphase

Purpose: Converts data from Real–Imaginary to Magnitude–Phase(degrees) format.

Output Format: Magnitude–Phase(degrees).

Usage: out_var = magphase(input_arg);

Code:

```
function retval = magphase (A)
  PX = 57.2957795130823;

if (size(A,2)==1)

         for idx = 1:size(A,1)
             v = A(idx,:);
             s11m = sqrt(real(v(1))*real(v(1))+imag(v(1))*imag(v(1)));
             s11p = PX*atan2(imag(v(1)),real(v(1)));

             y(idx,:) =  [s11m s11p];
         end
             retval = y;

else if (size(A,2)==3)

        for idx = 1:size(A,1)
            v = A(idx,:);
            s11m = sqrt(real(v(2))*real(v(2))+imag(v(2))*imag(v(2)));
            s11p = PX*atan2(imag(v(2)),real(v(2)));
```

```
            y(idx,:) =  [A(idx,1) s11m s11p];
        end
            retval = y;

else if (size(A,2)==5)

      for idx = 1:size(A,1)
            v = A(idx,:);
            s11m = sqrt(real(v(2))*real(v(2))+imag(v(2))*imag(v(2)));
            s11p = PX*atan2(imag(v(2)),real(v(2)));
            s21m = sqrt(real(v(3))*real(v(3))+imag(v(3))*imag(v(3)));
            s21p = PX*atan2(imag(v(3)),real(v(3)));
            s12m = sqrt(real(v(4))*real(v(4))+imag(v(4))*imag(v(4)));
            s12p = PX*atan2(imag(v(4)),real(v(4)));
            s22m = sqrt(real(v(5))*real(v(5))+imag(v(5))*imag(v(5)));
            s22p = PX*atan2(imag(v(5)),real(v(5)));

            y(idx,:) = [A(idx,1) s11m s11p s21m s21p s12m s12p s22m s22p];
     end
          retval = y;
    end
  end
end
```

A.16 Example

This section explains the stepwise execution of NetA for TL calibration and de-embedding. This example uses the long and short lines as line and through standards and de-embedds the s-parameters for the long line. The commands to be executed at the MATLAB prompt are:

```
get_mag_ang;

[thru,line,thru_o,line_o]
=touchstone(thru_data,line_data,freq_vect);

ldiff = 0.006;

c0 = 1.2e-10;

[serror,gamma_array,zc] =
tl_calib(thru,line,freq_vect,ldiff,c0,thru_o,line_o);

dut = xlsread(['long.xls']);

final_dut = final(dut);

deemb = tsl(serror,final_dut);
```

This will obtain the de-embedded result in the parameters *deemb*. To further convert it to the Magnitude–Angle format:

```
d_result = magphase(deemb);
```

Similarly, other utility routines can also be used by assigning the result to any temporary variable at the prompt and passing the right input argument.

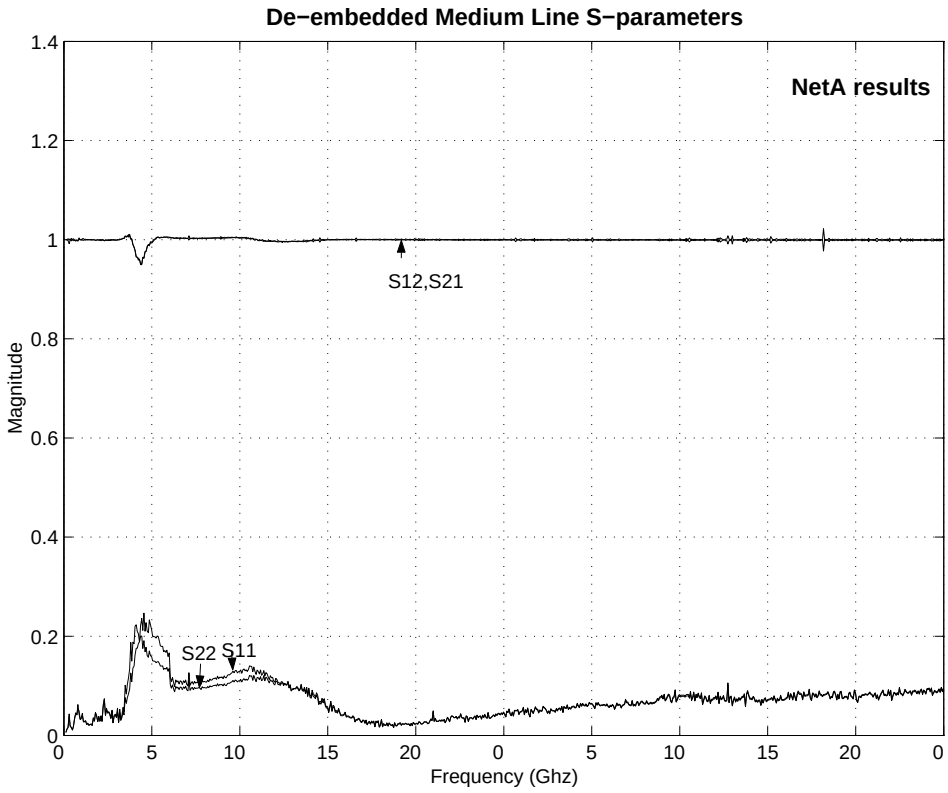


Figure A.1: De–embedded Medium line S-parameters (Magnitude)using the long and medium line as the line and through standards.

The medium and long line have been used as the Through and Line calibration standards. Raw parameters of the medium line have been used as the DUT. The derived parameters show the characteristics of a *Through* connection.

Appendix B

LabVIEW User Manual

This section is a user's manual for the LabVIEW Implementation of NetA. Fig. B.1 shows the front panel for the final VI that implements all of the four blocks explained in Chapter 5 in cascade.

1. The menu of the screenshot shows an arrow. As soon as you press that arrow button, the VI starts running and passes data from the front panel to the VI.

2. Press the first yellow button that reads *Grab Data*. It will immediately turn green. It will open up the current directory. Select the right raw data text files, i.e. long.txt, short.txt and freq.txt. These are made available from the VNA. After these files are included, the button will turn off and will wait for you to press the next button.

3. The interface between the modules in the VI is waiting for the user to ask it to perform the propagation constant, characteristic impedance determination and error matrix calculation. Likewise, press the second button that reads *Get_error_matrix*. It will turn green too and once the calculation will be completed, it will turn off.

4. Press the third button that reads *Get_dut_param*. Include the file that has the raw data for the DUT to be de-embedded.

5. Finally, press the *TSL–De–embedding* button and after calculation of the de-embedded data it will turn off. Data will appear after each of the four buttons turn off, in their respective indicators.

6. Also, the arrow button that was pressed in the beginning will now indicate completion of execution.

Data is now ready to be examined in each of the indicators. These indicators are just like two–dimensional arrays.

B.1 Installation Guide

The guidelines are as follows:

1. All files for the LabVIEW Implementation have been zipped under the name *TL_Labview*.

2. Install a version of LabVIEW that supports MATLAB Scripts.

3. You must have MATLAB and LabVIEW Installed on your workstation. It will work only on a Windows Operating System.

4. Unzip all files in your directory.

5. All MATLAB Scripts that you extract, must be included in the \Matlab\bin\win32 directory. These are internally called subroutines from the MATLAB scripts within LabVIEW.

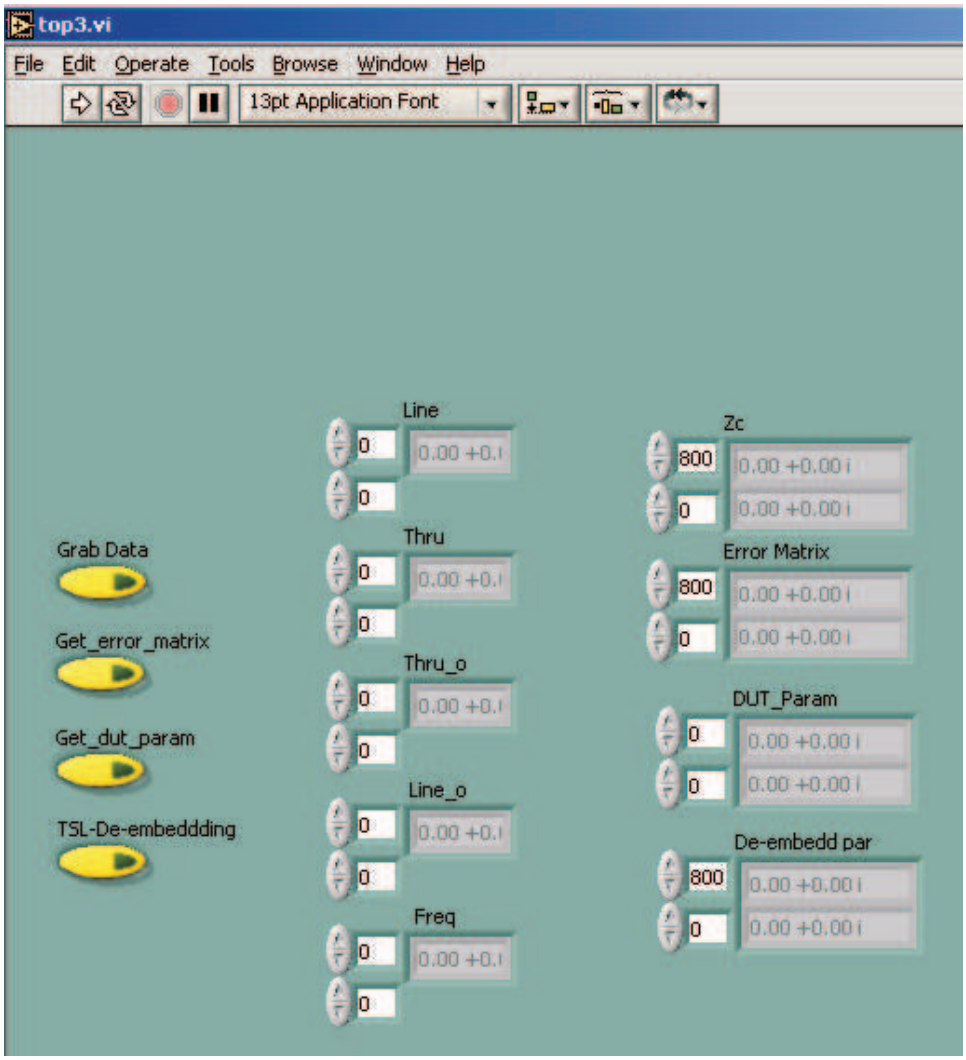


Figure B.1: Final Front Panel for TL calibration